
ATARI
ST
COMPUTER
Die Fachzeitschrift für den ATARI-ST Anwender.
Januar '87
ÖS 53,—/Sfr. 6,—
DM 6,—
1
Sprachen der künstlichen Intelligenz
XLISP & Cambridge LISP
Adimens ST
eine echt relationale Datenbank
Signum!
Textverarbeitung in einer neuen Dimension
Easyprommer
Nicht nur EPROMs leicht gebrannt
Messe-News aus den USA
Comdex '86

ATARI ST

Grafik vom Feinsten!

»monoSTar«

„Vergessen Sie alle Grafik- und Grafik-Hilfsprogramme, mit »monoSTar« haben Sie alles in einem, und noch mehr..." Komplett unter GEM-4 Arbeitsebenen, davon eine DIN A4-Freihandzeichnen (mit autom. Glättung), Punkte, Linien, Ellipsen, Kreise, Quadrate, Rechtecke, Polygone, Text, Radiergummi mit bel. Größe, Sprühdose, grobe und feine Lupe.
Beliebige Bildausschnitte können fließend bewegt, kopiert, gespiegelt, invertiert, gedehnt, gestaucht, geboten, ausgedruckt oder als Pinsel benutzt werden.
Bibliotheken erstellen und viele weitere Features...

Programm & Handbuch in Deutsch!

DM 99,–

»colorSTar«

Wie »monoSTar«, aber für Farbbetrieb...

Erweiterte Funktionen:
Trommel, Verforme, Schmiere, 50 Farbpaletten, Farbanimation mit 10 000 Schritten und 512 Farben, mischen von benachbarten Farben möglich (16 450 Farben!!). Natürlich auch unter GEM und in Deutsch!

Software made in Germany

DM 99,–

GFA Systemtechnik GmbH

Heerdter Sandberg 30
D-4000 Düsseldorf 11
Telefon 02 11/58 80 11





Rückblick nach vorn

Liebe Leserin, lieber Leser,

es ist soweit: Die Zeitschrift ST-Computer feiert ihr einjähriges Bestehen.

Für uns ist dieser erste Geburtstag ein Anlaß, uns bei Ihnen für Ihre Treue recht herzlich zu bedanken. Dieser Dank richtet sich ganz besonders an all jene Leser, die durch Einsendung ihrer Programme einen wesentlichen Teil zum Inhalt dieser Zeitschrift beitrugen und die thematischen Schwerpunkte auf diese Weise beeinflußten.

Bei dieser Gelegenheit möchten wir Ihnen endlich einmal die gesamte Mannschaft der ST-Computer vorstellen. Leider gelang es uns wegen terminlicher und organisatorischer Schwierigkeiten nicht, alle Mitarbeiter gemeinsam abzulichten. Unser Foto zeigt (von links) Marcelo Merino, Harald Schneider, Peter Gebhart, Jürgen Leonhard, Harald Egel, Heike Fuchs, Markus Nerding, Ulrike Seiler, Klaus Heuer, Uli Eickmann und Uwe Bärtels.

Gemeinsam haben wir uns im vergangenen Jahr stets bemüht, Ihnen einen aktuellen Überblick über das Geschehen rund um den ST zu liefern. Zudem wollten wir Ihnen praktische Tips und Hilfen zur Anwendung von Soft- und Hardware an die Hand geben. Bei der Themenauswahl suchten wir einen Mittelweg zwischen Anfängerproblemen und GEM-Akrobatik.

Das war manchmal nicht ganz leicht, und wenn es nicht immer zur Zufriedenheit aller Leser gelang, bitten wir Sie um Verständnis – konstruktive Kritik ist uns jederzeit willkommen.

Auch in Zukunft werden wir uns bemühen, diese Zeitschrift noch attraktiver zu gestalten – und noch näher an den Interessen unserer Leser zu orientieren. Dazu hoffen wir auch weiterhin auf Ihre Beteiligung.

Ein wunderschönes neues Jahr wünscht Ihnen

Ihre Redaktion

Inhalt

Allgemeines

Software

Hardware



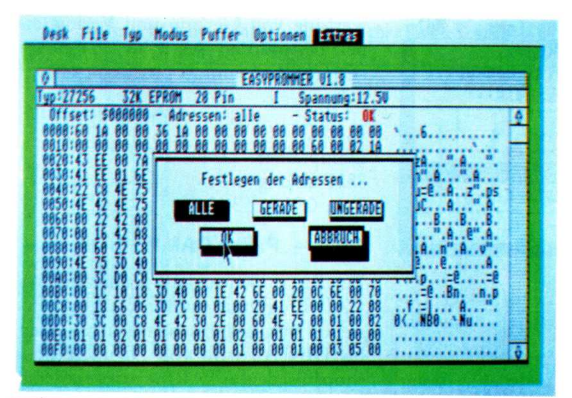
Adimens ST
Mit Adimens ST wird in dieser Ausgabe eine relationale Datenbank der mittleren Preisklasse vorgestellt. Adimens ST ist nicht nur sehr leistungsfähig, es bietet dem Anwender viele Vorteile gegenüber den bisher erhältlichen Programmen und läuft zudem in farbiger und monochromer Auflösung. 114

XLISP
Daß eine Programmiersprache nicht immer teures Geld kosten muß, beweist das Beispiel XLISP. Für alle, die sich einmal mit einer Sprache der Künstlichen Intelligenz vertraut machen wollen, besteht mit der Public-Domain-Version von LISP ein Angebot, an dem Sie kaum vorbeigehen können. 41



Relax
Wie in jedem Heft berichten wir über brandneue Spiele für den ST. Die Neuigkeit des Monats ist der langersehnte Flugsimulator von Sublogic, der einiges mehr zu bieten hat als seine Vorgänger-Versionen mit acht Bit. Aber auch Worldgames, Karate und Joust müssen sich keineswegs verstecken. 109

Inhalt



Easyprommer
Wir stellen unseren neuen EPROM-Brenner vor. Dabei wird ausführlich darauf eingegangen, was man über EPROMs und das „Brennen" von diesen Speicherchips wissen muß. Denn wer schätzt nicht die Vorteile, seine täglich verwendeten Programme jederzeit ohne lästiges Laden im Speicher zu haben? 10

Signum!
Textverarbeitungsprogramme für den ATARI ST gibt es mittlerweile schon eine ganze Menge. Mit „Signum!" erscheint nun das erste Programm, das sich auch hervorragend für wissenschaftliche Texte eignet. Es verfügt über einen eigenen Zeichensatz-Editor, so daß man nach Belieben Schrift, Zeichen und Formeln erstellen kann. 74



Comdex '86
Mit erwartungsvollen Blicken schaut man nach Amerika, da dort erfahrungsgemäß alle Produkte etwas früher erscheinen als bei uns. Welche Produkte dort vorgestellt wurden und ob atemberaubende Neuigkeiten darunter waren, erfahren Sie auf Seite: 16

Grundlagen

Kurse

Aktuelles

BACKUP für Harddisk

Ein sehr hilfreiches Utilit -Programm für alle Atari-Anwender, die eine Harddisk besitzen, wird von **Microdeal** angeboten. „Backup" ermöglicht die Herstellung einer Kopie von Files der Harddisk auf eine Floppy. Das Programm, das völlig GEM-unterstützt ist, bietet eine Reihe von Möglichkeiten, die das Herstellen einer Sicherheits-Kopie vom Inhalt einer Harddisk erleichtern.

Möchte man ein File kopieren, das größer ist als die vorhandene Kapazität einer Diskette, so fordert das Programm einen Disketten-Wechsel. Zudem besteht die Möglichkeit, bei erneutem Back Up nur diejenigen Programme zu kopieren, die beim ersten Back Up noch nicht vorhanden waren. Eine andere Möglichkeit für ein gezieltes Back Up ist die Option Back Up nach Datum. Dabei werden nur die Files kopiert, die an einem bestimmten Datum hergestellt wurden.

Möchte man ein Back Up von mehreren Disketten auf die Harddisk erstellen, hilft ein zusätzliches Programm, das auf der Diskette erhältlich ist.

Der Deutsche Vertrieb dieser Back-Up-Utility erfolgt über die Firma:

G. Knupe GmbH & Co KG
Postfach 354
4600 Dortmund 1
Tel.: 02 31 / 52 75 31

ATOMIUM: Darstellung von Molekülstrukturen am ST

Ein Programm, das die grafische Darstellung von Molekülstrukturen nach dem Kalottenmodell zeigt, wird nun für alle Rechner der ST-Serie angeboten.

Das Programm könnte Hilfen bieten für den Chemieunterricht an Schulen oder Universitäten, die sich mit der Veranschaulichung von Molekülstrukturen befassen. ATOMIUM ist eine Kombination von Datenbank- und komfortablen Grafikprogrammen. In der Datenbank sind bereits zahlreiche Informationen über Molekülstrukturen enthalten, so daß der Anwender lediglich eine Strukturformel angeben muß und anschließend die Molekülverbindungen grafisch auf dem Bildschirm ausgegeben werden. Der sogenannte Moleküleditor erlaubt das Zeichnen von neuen Molekülverbin-



dungen, die anschließend in der Datenbank abgespeichert werden können.

Das Programm, das vollständig unter GEM arbeitet, kostet DM 149,–.

Michael Schaumburg
Halemweg 21
1000 Berlin 13
Tel.: 0 30 / 382 31 05

ST Aktionär

Ein Programm, mit dem man die Kurse von Aktien an der Börse verfolgen kann, wird von der Firma E & C angeboten. Es nutzt die gesamten Merkmale der GEM-Umgebung voll aus. Mit der Maus lassen sich verschiedene Funktionen anwählen. Der Kurs von einer oder mehreren Aktien kann grafisch auf dem Bildschirm oder Drukker ausgegeben werden. Auch die Auswertung von Gewinn und Verlust einer bestimmten Aktie in einem gegebenen Zeitraum ist problemlos möglich.

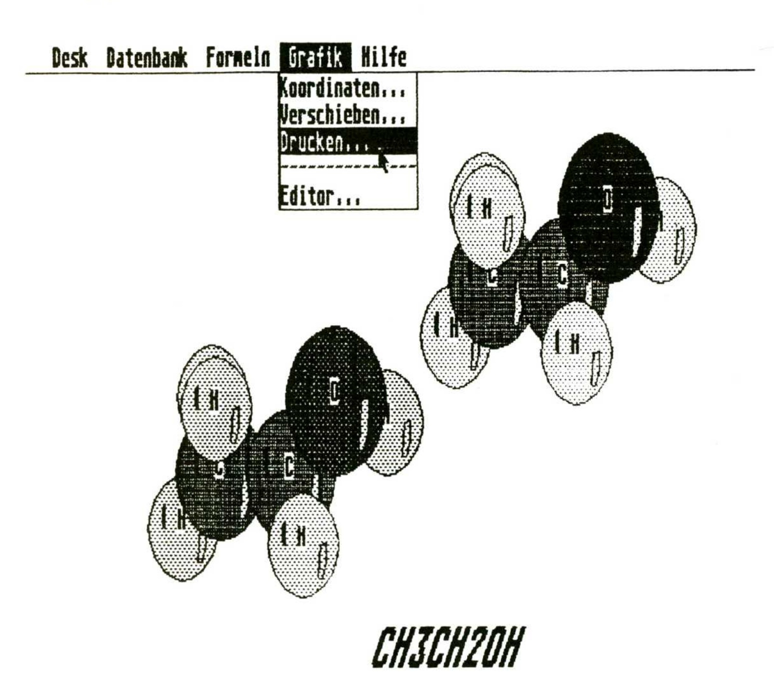
E & C
Dompfaffstr. 127a
8520 Erlangen
Tel.: 0 91 31 / 3 10 98

CAD – Paket CAMPUS

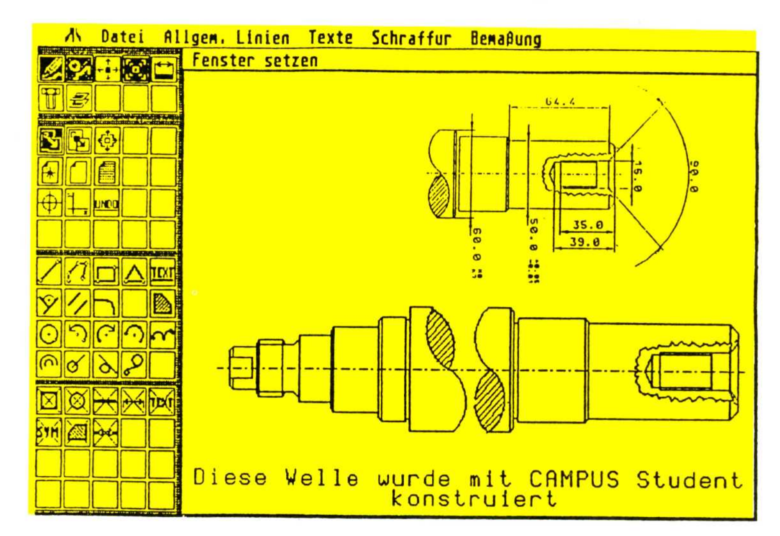
Ein in Deutschland entwickeltes CAD-Programm zum interaktiven Erstellen von technischen Zeichnungen bis DIN A0 wird von Digital Workshop für den ATARI ST angeboten.

Durch diverse Zeichenfunktionen (wie automatisiertes Bemaßen und Schraffieren) sowie durch die Integration verschiedener Konstruktionshilfen aus der Symbol- und Layouttechnik ist ein schnelles und komfortables Arbeiten möglich. Auch branchenspezifische Anwendungen können auf Wunsch in **CAMPUS** eingebunden werden. Die Aufwärtskompatibilität der Daten zu späteren Versionen ist gewährleistet. Das Programm kostet DM 798,–; eine Demoversion gibt's für DM 20,–.

Digital Workshop
Kornharpenerstr. 122a
4630 Bochum 1
Tel.: 02 34 / 31 13 04



Kalottenmodell für Molekülstrukturen

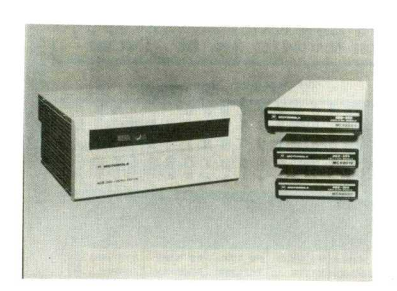


Low – Cost – CAD – System
CAMPUS – Student

Echtzeit-Emulation für die M68000-Familie

In Anbetracht der Nachfrage nach immer leistungsfähigeren Design-Tools für die M68000-Familie hat Motorola jetzt die Hard- und Software-Entwicklungsstation HDS-300 mit einem umfassenden Support für die leistungsfähigen Prozessoren der M68000-Serie ausgestattet. Die Entwicklung und der Test von Systemen wird wesentlich effizienter dargestellt, da der verwendete schnelle Emulationsspeicher – in Zusammenhang mit MC68020-Applikationen – die Echtzeit-Emulation bei 16 Mhz ohne Wartezyklen, bei 20 Mhz mit einem Wartezyklus ermöglicht.

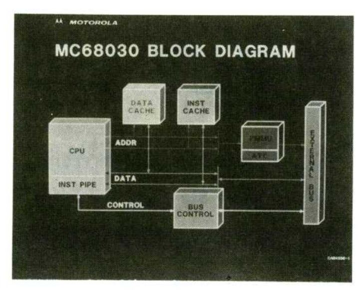
In Verbindung mit einem geeigneten Hostcomputer verfügt der Anwender mit der Entwicklungsstation HDS-300 über ein leistungsstarkes Hilfsmittel, mit dem Hardware und Software für all jene Systeme entwickelt werden können, die auf Prozessoren der Serie M68000 basieren.



Neue CPU 68030 von Motorola

Nach der neuen Versi der CPU 68020, die mit einer Frequenz von 25 MHz arbeitet, stellte Motorola nun einen neuen 32-Bit-Mikroprozessor vor. Der 68030 ist ein 32-Bit-Prozessor der zweiten Generation, bei dem eine Memory Management Unit bereits im Chip integriert ist. Der Prozessor besitzt zwei je 256 Bytes große Cache-Speicher, einen für Befehle und einen für Daten. Die Umrechnung von virtuellen in physikalische Adressen wird in einem Sonder-Speicher ausgeführt. Dadurch erfolgt die Adressierung sehr schnell. Der normale Buszugriff entspricht drei Taktzyklen und kann in einem speziellen Modus auf zwei Zyklen gekürzt werden.

Motorola GmbH
Arabellastr. 17
8000 München 81
Tel.: 0 89/ 92 72-0



Achtung – Berichtigung

Der **Mark-Williams C-Compiler** der bei der Firma PHILGERMA **498,–** DM kostet, wurde in der letzten ST-Computer irrtümlich mit 398,– DM abgedruckt. Dieser C-Compiler wird jetzt mit verbessertem EMACS-Editor ausgeliefert, der auch Texte mit deutschen Umlauten editieren kann.

GFA BASIC-Paket

Aus wettbewerbsrechtlichen Gründen ist GFA Systemtechnik gezwungen, das Paketangebot (bestehend aus GFA BASIC-Interpreter und Compiler zu DM 269,–) zurückzuziehen.

Wer dieses Paket schon bestellt haben sollte, muß damit rechnen, daß die Bestellung zurückgewiesen wird.

Weiterhin sind beide Produkte, also BASIC-Interpreter (Version 2.0 für DM 169,–) und Compiler (DM 169,–), einzeln zu beziehen.

Schade um das schöne Angebot.

GFA Systemtechnik
Heerdter Sandberg 30
4000 Düsseldorf 11
Tel.: 02 11 / 58 80 11

Lektor Programm von CCST

Im November-Heft hatten wir über ein neues Produkt der Firma CCST-Software berichtet. Leider ist uns bei der Kurzbeschreibung dieses Lektor-Programmes ein Fehler unterlaufen. Wir schrieben, daß dieses Programm unbekannte Worte nicht in die eingebaute Bibliothek übernehme. Das stimmt nicht: Das Programm übernimmt diese unbekannten Worte in die vorhandene Bibliothek, korrigiert sie jedoch nicht im Text, sondern gibt eine Liste der fehlerhaften Worte aus. Wir entschuldigen uns für dieses Mißverständnis bei unseren Lesern und dem Hersteller dieses Programmes.

Die Redaktion

CCST-Software, Tel. 0 84 63 / 93 43
Postfach 70, 8547 Greding

Lohnsteuer-Jahresausgleich mit dem ST

Ein Programm, das die Lohn- oder die Einkommensteuer für das Jahr 1986 berechnet und das auf allen ATARI-ST-Rechnern lauffähig ist, wird von **Tax-Software** angeboten.

Das Programm, das völlig im GEM eingebunden ist, wird in zwei Versionen geliefert. Eine ist für den privaten Gebrauch, die andere Version für den professionellen Anwender. Die erste Version kostet 79 DM und wird jedes Jahr gegen einen kleinen Aufpreis auf den neuesten Stand gebracht. Die Profi-Version besitzt eine kleine Datenbank, die die Ablagen von einzelnen Kunden-Dateien zur späteren Verarbeitung ermöglicht. Die Profi-Version kostet DM 159,–.

Tax-Software
Postfach 10 07 01
4040 Neuß 1

Neue Textutilities

Zwei neue Programme, die im Bereich der Textutilities anzusiedeln sind, werden von der Software-Firma **Pahlen & Krauss** angeboten. Beide Programme arbeiten z. B. mit 1ST_WORD zusammen und erweitern dessen Möglichkeiten um ein Vielfaches.

PKS_MAIL ist eine Adressverwaltung, die speziell das Erstellen von Formularen in beliebiger Form (zum Beispiel Rechnungen, Adressaufkleber, Paketkarten) und von Serienbriefen unterstützt. Das Programm kann (als ACCESSORY) direkt von 1ST_WORD aus aufgerufen werden, so daß die erzeugten Formulare sofort weiter bearbeitet werden können. Da auch Adressendaten aus anderen Karteiprogrammen (DB-Master, etc.) benutzt werden können, ist bei bereits vorhandenen Dateien keine neue Eingabe der Adressen erforderlich.

Beim Arbeiten mit 1ST_Word wurde oft die Möglichkeit vermißt, häufig benutzte Texte auf eine Funktionstaste legen zu können. PKS_USERKEY löst dieses Problem und zeigt in einem speziellen Anzeigefenster die Belegung an, die jederzeit editiert, gespeichert oder gelesen werden kann. Ein eingebauter Druckerspooler, eine Desktopuhr und weitere Funktionen sind jederzeit abrufbar.

Pahlen & Krauss Software
Kolonnenstr. 28
1000 Berlin 62
Tel.: 0 30 / 782 69 06

ST in A.U.G.E.

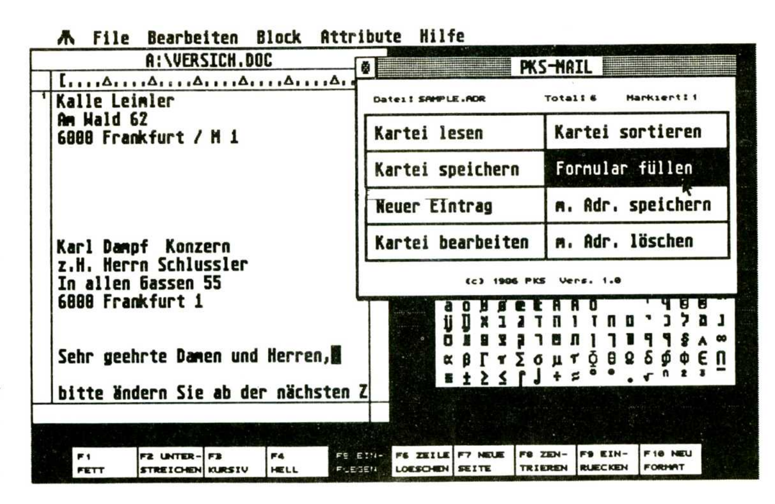
A.U.G.E., der größte eingetragene Computerclub Deutschlands, hat sich in Richtung ATARI ST 'geöffnet'. Dieser Club, der sich bis vor kurzem fast ausschließlich mit APPLE Rechnern beschäftigte, hat eine ATARI ST Arbeitsgruppe gegründet. Wenn Sie mitmachen wollen, wenden Sie sich an

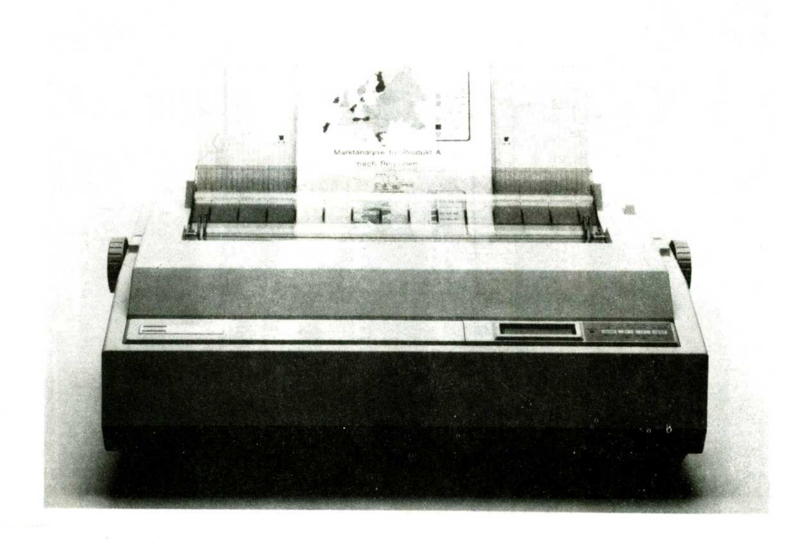
Wolfgang Wäsche
Bergerstraße 125
6000 Frankfurt 60
Tel.: 0 69 / 49 98 97

Fujitsu-Drucker DL 2600: Leise und schnell

Leise und benutzerfreundlich stellen sich die 24-Nadel-Drucker DL 2600 und DL 2600 Color von Fujitsu dar. Bei einem Geräuschpegel von nur 55 db(A) fliegen sie mit einer Geschwindigkeit von 125 Zeilen pro Minute über das eingelegte Blatt hinweg.

Die Benutzerfreundlichkeit zeigt sich besonders am Front-Panel mit dem 16-stelligen Display und dem danebenliegenden Bedienungsfeld. Über dieses Bedienungsfeld können denkbar einfach alle Betriebsfunktionen wie Schönschrift, Korrespondenzschrift, Schnelldruck, Grafik etc. eingestellt werden. Analog hierzu erscheinen die vorgenommenen Einstellunen auf dem LCD-Display und können vom Benutzer so-





fort überprüft und gegebenenfalls geändert werden.

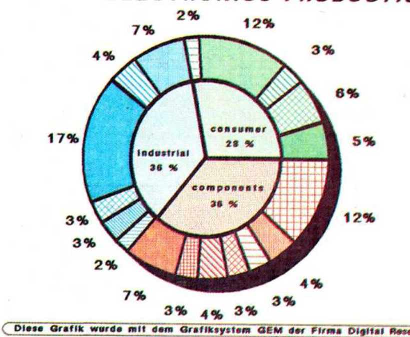
Durch den Mehrfarbdruck der C-Version und den umfangreichen Zeichensätzen eignet sich dieser Drucker nicht nur für Schriftverkehr und Kopien, sondern sogar für das Bedrucken von Overheadfolien.

Eine besondere Hilfe bei der täglichen Anwendung des Druckers ist der standardmäßig eingebaute Traktor, der den Wechselbetrieb von Einzelblatt und Endlospapier erlaubt. Wahlweise stehen parallele und serielle Schnittstellen zur Verfügung, was den Anschluß an jeden Computer erlaubt.

FUJITSU Deutschland GmbH
Rosenheimer Str. 145
8000 München 80
Tel.: 0 89 / 41 30 10

FUJITSU 24-Nadel Drucker

ELECTRONICS PRODUCTION IN JAPAN 1984



- Color TV sets
- Tape recorders
- Stereos
- VTR
- Other Products
- wired comm. equipm.
- Radio comm. equipm.
- Computers
- Otherass.electr.equipm.
- electr. measuring instr.
- electr. desktop calc.
- passivemech.electr.comp.
- mechanical parts
- System article parts
- other parts
- electr. tube
- semicond. devices
- integrated circuits

Diese Grafik wurde mit dem Grafiksystem GEM der Firma Digital Research erstellt.

Easyprommer
– mehr als ein EPROM-Programmiergerät

Wäre es nicht schön, ständig benötigte Accessories wie das Kontrollfeld, den VT52-Emulator oder eine RAM-Disk sofort nach dem Einschalten des Rechners parat zu haben? Oder wie wär's mit einer Textverarbeitung oder dem GFA-BASIC auf einer EPROM-Karte – das Laden solcher oder anderer häufig benutzer Programme könnte in Zukunft entfallen. Vielleicht stören Sie sich auch an einigen Fehlern des Betriebssystems (TOS) oder Sie wollen Programme von Diskette doppelt so schnell laden wie bisher. Neugierig geworden? Dies sind noch längst nicht alle Möglichkeiten, die Ihnen der Easyprommer bietet. Dieser Artikel soll Ihnen nicht nur die vielfältigen Möglichkeiten des Easyprommers, sondern auch die grundlegende Technik eines EPROMs und eines Programmiergerätes beschreiben.

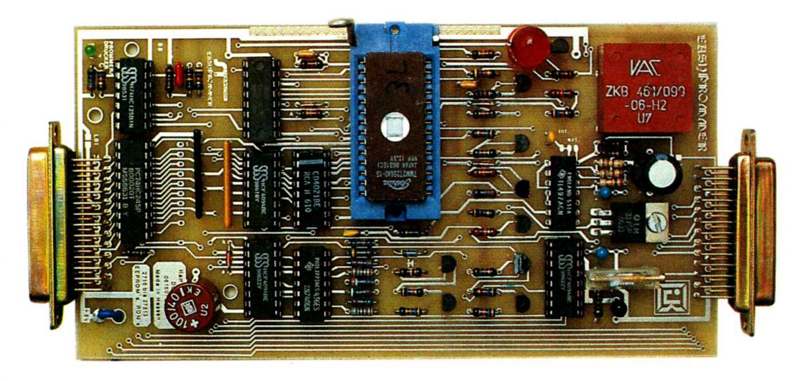


Bild 1: Der Easyprommer

Was ist ein EPROM?

Um dies verständlich zu machen, möchten wir ein wenig ausholen und das Prinzip eines Rechners erklären. Abbildung 1 zeigt das Schema eines Rechners: Er besteht aus der Zentraleinheit mit der CPU, der Speichereinheit und der Ein- und Ausgabeeinheit, mit der eine Verbindung zur Außenwelt geschaffen wird. Die Speichereinheit ist dabei unterteilt in einen Festwertspeicher und einen Speicher mit veränderbaren Inhalt (RAM). Beim ATARI ST ist der RAM-Speicher, mindestens 512 KByte groß. Der Festwertspeicher, in dem das Betriebssystem (TOS), falls vorhanden, in sechs ROMs oder auch EPROMs untergebracht ist, hat eine Kapazität von 192 KByte. Dieser Festwertspeicher kann durch eine externe EPROM-(ROM)-Karte um weitere 128 KByte erweitert werden, so daß insgesamt 320 KByte zur Verfügung stehen. Durch einen Trick läßt sich der externe Speicher sogar noch weiter vergrößern. Doch dazu später.

Im RAM-Speicher (**R**andom **A**ccess **M**emory-Speicher mit beliebigem Zugriff) werden im allgemeinen die Programme, Daten und Variablen gespeichert. Es ist also ein Speicher, den man lesen und beschreiben kann. Dieser Speicher „vergißt" aber seine Daten, sobald er nicht mehr mit Strom versorgt wird, also wenn der Rechner ausgeschaltet wird.

Zur Gruppe der Festwertspeicher gehört das EPROM (**E**raseable **P**rogrammable **R**ead **O**nly **M**emory). Zu deutsch: Löschbarer (und wieder) programmierbarer Nur-Lese-Speicher. Ein EPROM läßt sich also mit Hilfe eines Programmiergerätes wie z. B. dem Easyprommer programmieren. Ein so programmiertes (in der Umgangssprache: gebranntes) EPROM behält seine Daten (Programm/e) über Jahre, **ohne** daß dazu eine Betriebsspannung notwendig ist. Ein Löschen des Speicherinhaltes ist nur durch Bestrahlung mit ultraviolettem Licht möglich. Dazu besitzt jedes EPROM ein kleines Fensterchen, durch das Licht auf das Silizium-Plättchen mit den zigtausend Transistoren dringen

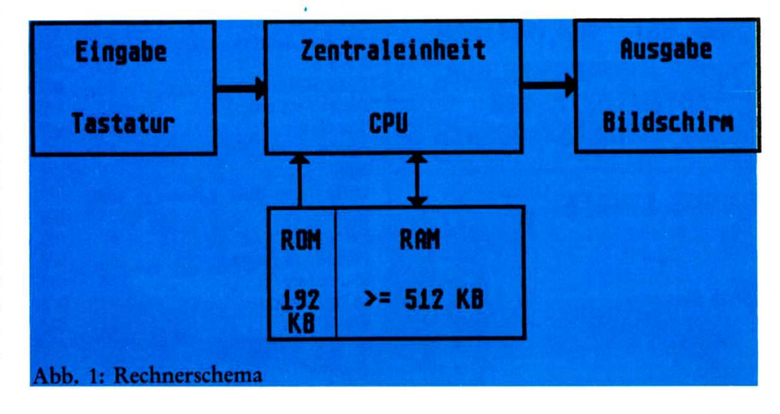


Abb. 1: Rechnerschema

Sonderheft

für alle ST-User



Das ST-Sonderheft bietet unter anderem folgende Themen:

- **Compilertechnik leicht gemacht**
  Die komplette Anleitung zum Programmieren eines Pseudo Pascal Compilers mit Grundlagen und Technik

- **Einführung in Prolog**
  Alles über die Sprache der künstlichen Intelligenz Prolog. Eine Einführung für Einsteiger und Fortgeschrittene mit vielen Beispielen.

- **3 D Graphik**
  Ein Programm zur Erstellung von Rotationskörpern. Diese Körper kann man konstruieren, vergrößern, verkleinern und rotieren lassen.

- **Trickfilm**
  Ein Programm zur Erzeugung von kontinuierlichen Bewegungsabläufen.

- **Superschnelle Apfelmännchen**
  Erzeugen bis zur Ausschnittsvergrößerung ist kein Problem. **Der Computer als Psychologe:** Mit Eliza, dem Programm für künstliche Intelligenz. **Eine Super 68000 Assembler Bibliothek:** Routinen einfach aufrufen und effizient programmieren. **Ray Tracing Programm:** Mit diesem Strahl-Verfolgungsprogramm erzeugen Sie in normalen Bildern (decas) phantastische Graphiken und Spiegelungen (s. Titelbild). Sowie Bericht über **Monitore und Drucker** für den Atari und über den Stargast **Commodore Amiga.**

- **Fensterverwaltung**
  Ein Superprogramm in **GFA-BASIC.**

Bestell-Abschnitt

Einsenden an: Heim-Verlag · Heidelberger Landstr. 194 · 6100 Darmstadt-Eberstadt

Senden Sie mir bitte ______ Stück des ST-Sonderheftes

Ich zahle keine Versandkosten. Den Betrag für das Sonderheft begleiche ich durch

☐ beigefügten Verrechnungsscheck

☐ Überweisung auf Konto 4 000 625 Sparkasse Darmstadt

Name

Straße

Ort

schließend sofort ausgedruckt werden kann. Außerdem stört der Easyprommer die Arbeit mit einer Textverarbeitung nicht. Im Gegenteil, durch den eingebauten Treiber werden die vom Rechner **ungepufferten** Druckersignale verstärkt, so daß auch ein extrem langes Druckerkabel oder niederohmige Pull-up-Widerstände im Drucker nicht mehr stören. Als Spannungsversorgung benötigt der Easyprommer nur eine positive 5-Volt-Spannung, die an einer der beiden Maus/Joystick-Ports abgenommen wird.

Die Hardware

Die Schaltung des Easyprommers ist größtenteils in CMOS-Technologie realisiert worden, so daß der Stromverbrauch nur maximal 400 mA (kurzzeitig) beträgt. Dieser Strom kann problemlos allen ST-Modellen entzogen werden. Das Blockschaltbild des Easyprommers ist in Abbildung 3 zu sehen. Der Easyprommer kann an jeder „normalen" Centronics-Schnittstelle betrieben werden, die folgende Signale zur Verfügung stellt:

Pin 1 Strobe (nur Ausgang)
Pin 2 D0 Datenleitung
. .
. .
. .
Pin 9 D7 Datenleitung
Pin 11 Busy (nur Eingang)

Somit ist die Hardware des Easyprommers relativ unabhängig vom Rechner und könnte ebenso z. B. am Amiga betrieben werden. Die entsprechende Software für diesen Rechner ist übrigens in Arbeit. Obwohl der Easyprommer am Parallelport arbeitet, erfolgt die eigentliche Datenübertragung vom und zum EPROM seriell. Durch die hohe Taktfrequenz des ATARI ST ist auch bei serieller Übertragung eine enorme Geschwindigkeit zu erreichen. Über das Busy-Signal (Pin 11) werden die Daten aus dem EPROM in den Rechner gelesen, und über die Datenleitung D7 (Pin 9) werden die Daten zum EPROM geschickt. Das Strobe Signal (Pin 1) des Parallel-Ports dient als Übernahme-Signal der Daten und die verbleibenden sieben Datenleitungen D0 bis D6 werden als Takt-, Zähl- und Rücksetzleitungen benötigt.

Auf der Platine befinden sich zwei Leuchtdioden. Die rote leuchtet immer dann, wenn in irgend einer Form auf das EPROM zugegriffen wird. Solange diese Anzeige leuchtet, sollte der Sockelzustand nicht verändert werden. Die grüne Leuchtdiode dient als Indikator für den Drucker. Leuchtet sie, so kann problemlos über einen angeschlossenen Drucker gedruckt werden, anderenfalls befindet man sich im „Prommer-Modus", in dem nicht gedruckt werden kann.

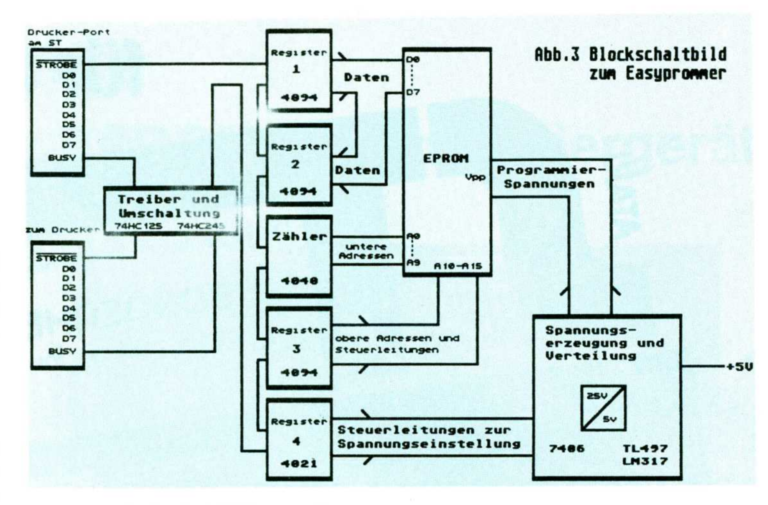


Abb. 3: Blockschaltbild zum Easyprommer

Zum Einstellen der Programmierspannung – diese variiert bei den einzelnen EPROM-Typen zwischen 12, 21 und 25 Volt – ist ein Spindel-Poti (Einstellregler) vorhanden. Für Bastler, die sich den Easyprommer als Bausatz bestellen, ist dies der einzige Abgleichpunkt, der allerdings mit einem gewöhnlichen Voltmeter und dem mitgelieferten Test- und Einstellprogramm keine Schwierigkeiten bereitet. Das Fertiggerät wird abgeglichen und getestet geliefert.

Die Software

Um die universell ausgelegte Hardware auch vollständig ausnutzen zu können, benötigt man eine ausgereifte Steuersoftware. Mitgeliefert wird, neben dem eigentlichen Treiberprogramm zum Easyprommer, ein Install-Programm (RAM-Disk) und ein „Save-Programm", die alle sowohl in mittlerer als auch in hoher Auflösung funktionieren. Mit dem Install-Programm kann eine RAM-Disk mit unterschiedlichen, wählbaren Kapazitäten eingerichtet werden. Sie kann als normale RAM-Disk benutzt werden, dient aber im Zusammenhang mit dem Easyprommer dazu, jedes beliebige Programm oder auch Desk-Accessory in EPROMs zu brennen. Dazu kopieren Sie einfach die Programme, die später in die EPROMs gebrannt werden sollen, auf diese RAM-Disk. Anschließend müssen Sie das „Save-Programm" starten. Dieses Programm erzeugt nun aus den Programmen auf der RAM-Disk ein File auf Diskette, das schließlich mit dem Easyprommer in EPROMs bzw. EEPROMs gebrannt werden kann. Damit ist erstmalig die Möglichkeit geschaffen häufig benutzte Programme oder Accessories in EPROMs zu brennen, die beim Einschalten des Rechners automatisch von der EPROM-Karte gebootet werden.

Das eigentliche Treiberprogramm ist, wie es sich für den ATARI ST gehört, vollständig in GEM eingebunden, kann aber auch zum größten Teil über die Tastatur bedient werden. Das „Arbeitsfeld" erklärt sich durch Dialogboxen von selbst, so daß das ausführliche Bedienungshandbuch eigentlich nur für die besonderen Eigenschaften des Easyprommers zu Rate gezogen werden muß. Abbildung 4 zeigt die Drop-Down-Menüs mit allen Menüpunkten, die im folgenden näher beschrieben werden. Dadurch wird auch die Leistungsfähigkeit der Soft- und Hardware transparent:

Desk

Dieses Menü ermöglicht den Zugriff auf die im System installierten Programme mit der Extension .ACC (Accessories), wie z. B. das Kontrollfeld, die Druckeranpassung usw.

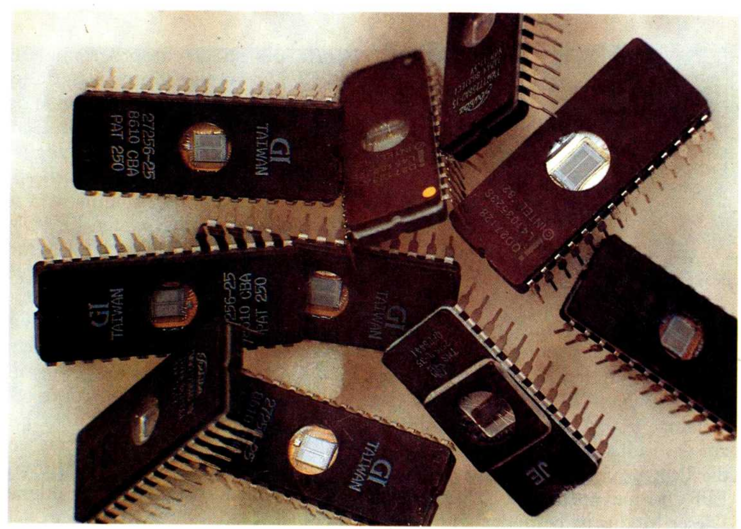


Bild 2: EPROMs

kann (siehe Bild 2). Durch das Licht werden sämtliche Speicherzellen auf High-Potential gesetzt. Aus diesem Grund enthält ein neues, leeres oder gelöschtes EPROM in jeder Zelle den Wert $FF (8-Bit-Organisation). Ein gelöschtes EPROM kann anschließend wieder programmiert werden.

Die Entwicklung der EPROMs

Waren vor einigen Jahren EPROMs mit einer Speicherkapazität von acht KByte kaum bezahlbar, so ist ein solches EPROM heute schon für etwa acht Mark zu erstehen. Zudem konnten die Speicherkapazitäten durch immer feinere Strukturen ständig erhöht werden. So ist ein EPROM mit 32 KByte Speicherkapazität (27256) gerade im Zusammenhang mit dem ATARI ST als Standard zu bezeichnen. Ein solches EPROM gibt es zur Zeit für ca. zwölf Mark in Elektronik-Fachgeschäften. Aber auch 64-KByte-EPROMs sind durchaus gebräuchlich.

Die äußerlich so ähnlichen Chips sind jedoch in ihrer Programmierung teilweise grundverschieden. Gab es vor einiger Zeit noch EPROMs, die zum Betrieb zwei oder drei Spannungen benötigten und die nicht kompatibel zu den Produkten anderer Hersteller waren, so haben sich heute glücklicherweise die günstigen und vielverwendeten EPROMs der 27er Serie durchgesetzt. Die Speicherkapazität errechnet sich bei diesen Typen aus der Zahlenfolge hinter der „27“. Wegen der Organisation des Datenbusses zu 8 Bit Breite muß die Zahlenfolge durch 8 geteilt werden und man erhält die Kapazität des EPROMs in KByte (1 Kilobyte = 1024 Byte). So besitzt z. B. das EPROM des Typs 27256 genau 262144 Speicherzellen, die zu 32768 ★ 8 Bit organisiert sind, dies entspricht dann 32 KByte (= 256/8 KB).

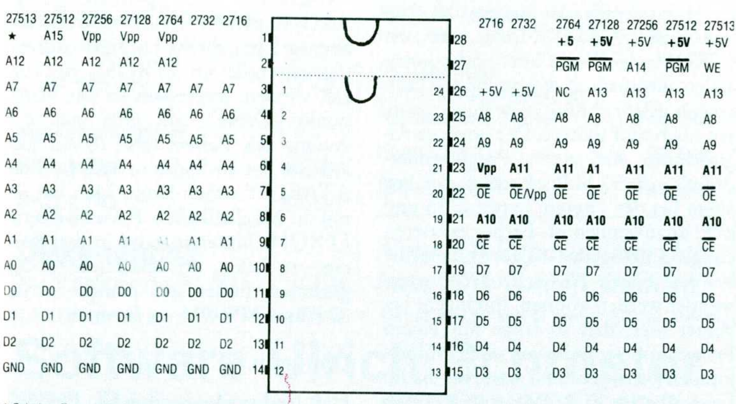
Die Anschlußbelegung

Bei den EPROM-Typen der 27er Serie haben die Hersteller durch eine einheitliche Anschlußbelegung die Möglichkeit geschaffen, bestehende Speicherplatinen (so auch die ROMs im ATARI ST) durch EPROMs zu ersetzen. Außerdem können diese EPROMs gegen statische RAMs oder gegen sogenannte **EEPROMs** (E^2PROMs) ausgetauscht werden. Dies sind den EPROMs ähnliche Speicherchips, bei denen die eingebrannten Daten nicht durch ultraviolettes Licht, sondern auf elektrischem Wege gelöscht werden. Diese modernen Bausteine können ebenfalls mit dem Easyprommer gelesen und programmiert werden. In der Abbildung 2 sind die gebräuchlichsten EPROMs und ihre Pinbelegung gezeigt.

Nachdem Sie nun wissen, daß EPROMs auch nur Speicher-Chips sind, wollen wir uns mit dem **Programmiergerät,** dem Easyprommer beschäftigen. Seine Elektronik ist auf einer doppelseitigen, durchkontaktierten und mit Lötstoplack versehenen Platine aufgebaut. Auf dieser Karte mit den Abmessungen 16,7 x 8,6 cm befinden sich alle Bauteile einschließlich der Spannungsversorgung und dem Programmiersockel, der ein leichtes Einsetzen der EPROMs ermöglicht.

Der Anschluß an den ST

Der Easyprommer wird einfach am Parallelport des ATARI ST (Druckerport) angeschlossen. Er kann wahlweise direkt eingesteckt oder über ein Verlängerungskabel neben dem ST betrieben werden. Bemerkenswert ist dabei, daß man einen angeschlossenen **Drucker weiterhin betreiben** kann, **ohne** daß ein Kabel umgesteckt werden muß. Der Easyprommer besitzt dazu einen Treiberbaustein samt Umschaltung, der sämtliche Signalleitungen des Parallelports verstärkt, und einen 25poligen Cannon-Stecker, an den der Drucker angeschlossen werden kann. Dies bietet den Vorteil, daß z. B. der Inhalt eines EPROMs mit dem Easyprommer gelesen und an-



27513	27512	27256	27128	2764	2732	2716					2716	2732	2764	27128	27256	27512	27513
★	A15	Vpp	Vpp	Vpp			1			28			**+5**	**+5V**	+5V	**+5V**	+5V
A12	A12	A12	A12	A12			2			27			$\overline{PGM}$	$\overline{PGM}$	A14	$\overline{PGM}$	WE
A7	A7	A7	A7	A7	A7	A7	3	1	24	26	+5V	+5V	NC	A13	A13	A13	A13
A6	A6	A6	A6	A6	A6	A6	4	2	23	25	A8	A8	A8	A8	A8	A8	A8
A5	A5	A5	A5	A5	A5	A5	5	3	22	24	A9	A9	A9	A9	A9	A9	A9
A4	A4	A4	A4	A4	A4	A4	6	4	21	23	**Vpp**	**A11**	**A11**	**A1**	**A11**	**A11**	**A11**
A3	A3	A3	A3	A3	A3	A3	7	5	20	22	$\overline{OE}$	$\overline{OE}$/Vpp	$\overline{OE}$	$\overline{OE}$	$\overline{OE}$	$\overline{OE}$	$\overline{OE}$
A2	A2	A2	A2	A2	A2	A2	8	6	19	21	**A10**	**A10**	**A10**	**A10**	**A10**	**A10**	**A10**
A1	A1	A1	A1	A1	A1	A1	9	7	18	20	$\overline{CE}$	$\overline{CE}$	$\overline{CE}$	$\overline{CE}$	$\overline{CE}$	$\overline{CE}$	$\overline{CE}$
A0	A0	A0	A0	A0	A0	A0	10	8	17	19	D7	D7	D7	D7	D7	D7	D7
D0	D0	D0	D0	D0	D0	D0	11	9	16	18	D6	D6	D6	D6	D6	D6	D6
D1	D1	D1	D1	D1	D1	D1	12	10	15	17	D5	D5	D5	D5	D5	D5	D5
D2	D2	D2	D2	D2	D2	D2	13	11	14	16	D4	D4	D4	D4	D4	D4	D4
GND	GND	GND	GND	GND	GND	GND	14	12	13	15	D3	D3	D3	D3	D3	D3	D3

* Bei alten Typen nicht belegt, bei neuen Typen Zurückschalten der Bank 0.

Abb. 2: Pinbelegung gebräuchlicher Eproms



Abb. 4: Die Drop-Down-Menüs

File

Dieser Menüpunkt erlaub Diskettenoperationen wie Laden, Speichern, Anhängen und Löschen von Dateien.

Typ

Hier kann unter den verschiedenen EPROM-, ROM- und EEPROM-Typen der gewünschte Typ ausgewählt werden. Erwähnenswert ist der Menüpunkt „**INT. ID**", der EPROMs der Firma Intel und kompatible automatisch identifiziert und den entsprechenden Typ sowie die Programmierspannung einstellt. Sollte ein EPROM eingesetzt werden, daß diese Möglichkeit nicht bietet, so wird ein Paritätsfehler ausgegeben. In diesem Fall müssen Sie vor dem Programmieren den jeweiligen Typ von Hand einstellen.

Ferner besteht die Möglichkeit, auch **ROM**s (z. B. die TOS-ROMs von ATARI) einzulesen. Dabei werden selbständig die entsprechenden Selektionssignale pegelmäßig ausgerichtet. Diesen Komfort bietet kaum ein anderer „Prommer".

Modus

In diesem Menü wird ein geeigneter Programmieralgorithmus gewählt. Neben dem „AUTO"-Modus, der bei EPROMs der Firma Intel und kompatiblen automatisch den am besten geeigneten Algorithmus auswählt, existieren noch fünf weitere. Der normale Algorithmus hat einen Programmierimpuls von 50 Millisekunden, der vor allem bei den älteren Typen 2716 und 2732 anzuwenden ist. Ferner existieren einige „intelligente" Algorithmen, die bei den Typen 2764A bis 27513 angewendet werden können. Intelligent bedeutet hier, daß so lange mit einem Programmierimpuls von einer Millisekunden (1ms) gebrannt wird, bis die Information im EPROM gespeichert ist. Anschließend erfolgt eine Sicherheitszugabe vom Vierfachen der bis dahin vergangenen Programmierzeit. Im zuletzt genannten Modus (1ms/4★) beträgt die gesamte Programmierzeit für ein 8-KByte-EPROM 45 Sekunden. Verringert man die Sicherheitszugabe auf das Einfache (1ms/1★), so beträgt die Programmierzeit für ein 8-KByte-EPROM nur noch etwa 20 Sekunden. Dieser Modus ist besonders zum Austesten von Programmen geeignet. Außerdem ist speziell für die modernen 64-KByte-EPROMs 27512 und 27513 ein extrem schneller Algorithmus vorhanden, der diese Typen in nur 30 Sekunden programmiert. Das 27513 ist übrigens ein „gebanktes" EPROM: Die 64 KByte dieses Typs setzen sich aus 8 Blöcken zu 8 KByte zusammen, die durch drei Steuerleitungen ausgewählt werden.

Puffer

Innerhalb dieses Drop-Down-Menüs stehen dem Anwender Befehle zur Verfügung, um die Daten im Speicher (Puffer) mit dem eingebauten Hex/ASCII-Monitor zu manipulieren. Dabei verhält sich der Monitor wie ein Fenster unter GEM, d. h. man kann sich z. B. den Inhalt des Puffers bequem mit Hilfe der Maus ansehen. Funktionen wie Füllen, Suchen und Verschieben von Speicherbereichen sind schnell und einfach zu erreichen, ebenso kann direkt zu einer Adresse (Speicherzelle im EPROM) gesprungen werden. Interessant ist der Menüpunkt „Offset", mit dem man den Anfang des Puffers (also 0) auf jede Adresse des verfügbaren Speichers des ATARI ST setzen kann. Man hat damit die Möglichkeit, z. B. vier 8-KByte-EPROMs hintereinander in den Speicher zu laden, um anschließend den gesamten Puffer auf einmal in ein 32-KByte-EPROM zu brennen.

Optionen

Hier findet man die üblichen Funktionen wie Lesen, Vergleichen, Leertest und Programmieren. Bit-Test überprüft, ob ein bereits programmiertes EPROM mit den Daten des Puffers programmiert werden kann, ohne daß es vorher gelöscht werden muß. Generell ist ein überbrennen nur möglich, wenn eine 1 zu einer 0 werden soll. Beim Menüpunkt Prüfsumme wird die Prüfsumme des eingesetzten EPROMs berechnet und angezeigt. Notiert man sich diese Zahl, erhält man einen für den Inhalt des EPROMs repräsentativen Wert, so daß eine spätere Verwechslung nicht möglich ist. Der Menüpunkt Fastprog. dient zum Programmieren von **EPROM-Simulatoren** oder **CMOS-RAMs**. Der Menüpunkt Einzelbyte erlaubt es, unter Angabe von Start- und Endadresse einzelne Bytes zu brennen. Hierzu ein kleines Beispiel:

Laden mit doppelter Geschwindigkeit

Vorausgesetzt, Sie besitzen das Betriebssystem des ATARI ST in ROMs, so können Sie sich das ROM mit der Bezeichnung U7 mit dem Easyprommer in ein EPROM des Typs 27256 kopieren. Anschließend können Sie das EPROM an die Stelle des ROMs wieder einsetzen und der Rechner müßte nach wie vor booten. Möchten Sie nun, daß Ihr Rechner sämtliche Programme doppelt so schnell von Diskette lädt, müssen Sie so verfahren: Tauschen Sie wieder das EPROM mit dem ROM aus und lesen Sie das EPROM mit dem Easyprommer in den Puffer ein. Verändern Sie nun den Wert des Puffers an Adresse $0DC7 von $14 auf $10 um und überbrennen Sie diese Zelle im Einzelbyte-Modus. Wenn Sie jetzt noch einmal das ROM U7 mit dem geänderten EPROM vertauschen, dann werden Sie beim Laden von Programmen eine Beschleunigung um Faktor 2 feststellen.

Extras

Unter dem Menüpunkt Adressen kann man bestimmen, ob man mit allen oder nur mit den geraden bzw. ungeraden Adressen arbeiten möchte (siehe Bild 3). Dieser 68000er-spezifische Adressierungsmodus gilt für alle Funktionen und ist gerade beim ATARI ST sehr wichtig. Beim ST sind alle Speicher-Bänke, egal ob RAM, ROM bzw. EPROM, in eine Low- und eine High-Bank unterteilt. Ein 16-Bit-Datenwort des Prozessors ist also immer in **zwei** EPROMs abgelegt, die unteren 8 Bit

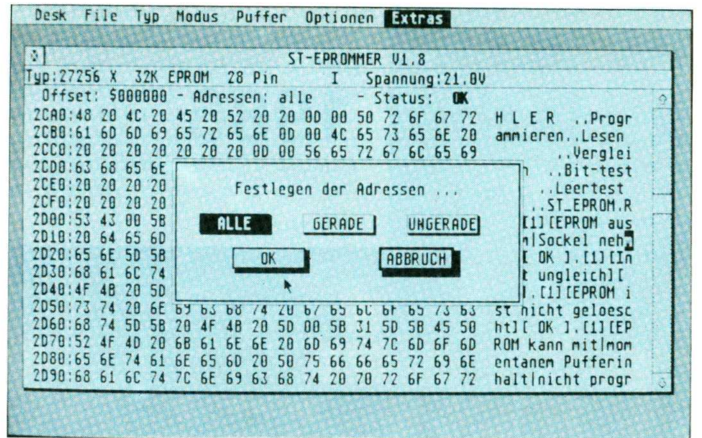


Bild 3: Menü: Extras

im EPROM der Low-Bank und die oberen acht entsprechend im EPROM der High-Bank. Alle geraden Adressen eines Programms entsprechen der High-Bank (upper) und die ungeraden Adressen befinden sich in der Low-Bank (lower).

Schließlich kann man mit dem Menüpunkt Drucken einen angegebenen Pufferbereich auf einen Drucker ausgeben.

Update der Software

Sollte in Zukunft ein neuer EPROM-Typ auf den Markt kommen, so wird dieser so bald wie möglich in der Software integriert werden – vorausgesetzt, daß keine Änderungen an der Hardware erforderlich sind. Dies ist jedoch wegen des universellen Konzepts unwahrscheinlich.

Die EPROM-Karte

Wie schon zu Beginn erwähnt, besteht die Möglichkeit, den Festwertspeicher des ST durch eine externe EPROM Karte um 128 KByte (4 EPROMs á 32 KB) zu erweitern. Eine solche „normale“ EPROM-Karte (-Bank) gibt es zur Zeit bei diversen Fachhändlern zu kaufen. Über die „ST-Computer“-Redaktion wird es Anfang 1987 eine „gebankte“ EPROM-Karte geben, die bis zur **vierfachen** Kapazität einer normalen Karte, nämlich **512 Kilobyte**, aufweist. Auf einer solchen Karte, die an den Modulschacht des ST eingesteckt wird, finden bis zu acht 64 KByte EPROMs Platz. Damit ist es möglich, auch riesige Programme, z. B. eine Textverarbeitung und/oder eine Programmiersprache, in EPROMs zu brennen und im „EPROM-Laufwerk“ ständig parat zu haben. Eine EPROM-Karte verhält sich bezüglich Bedienung und Geschwindigkeit ähnlich wie eine RAM-Disk, nur daß von ihr nur gelesen werden kann.

Lieferumfang und Bezugsmöglichkeit

Der Easyprommer ist als Fertiggerät und als Bausatz zu beziehen. Der Bausatz wird komplett mit Platine, allen Bauteilen und der Software einschließlich einem Test- und Einstellprogramm geliefert. Das Fertiggerät kostet DM 349,– und der Bausatz DM 298,–. Ein Gehäuse für das Gerät und ein Verlängerungskabel (zum Betrieb des Easyprommers neben dem Rechner) sind als gesondertes Zubehör erhältlich.

(UB)

Bezugsadresse:
ST Computer Redaktion
Schwalbacher Str. 64
6236 Eschborn
Tel. 0 61 96 / 48 21 58

Messebericht
Comdex Herbst 1986

Die Messe der Superlative? Für Atari und die diversen Software Häuser kann man diese Frage mit „Ja“ beantworten. Das Schwerpunktthema für Atari war auf dieser Ausstellung nicht die Hardware, sondern die Demonstration diverser Software Pakete. Es wurde zwar, wie auch auf anderen Messen vorher, der Blitterchip gezeigt, aber selbst in USA wurde zu einem definitiven Liefertermin keine Stellung bezogen.

Sogar eine deutsche Firma war durch Herrn Brockner aus dem Hause Print Technik vertreten, der hier zum ersten Mal seinen Video Digitizer der Öffentlichkeit in den Vereinigten Staaten präsentierte.

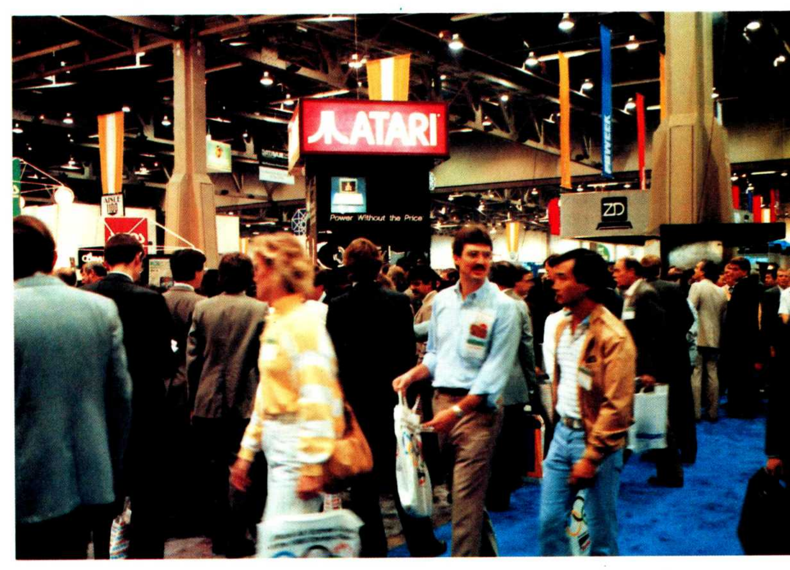


Bild 1: Ein Blick auf den ATARI-Stand

Desktop Publishing war nicht nur auf dem Atari Stand ein Zauberwort, sondern viele Anbieter der MS DOS Rechner versuchten auf diesem Gebiet etwas zu zeigen. Interessant war die Feststellung, daß bedienungsfreundliche Programme dieser Kategorie auf Rechnern der MS DOS Klasse unendlich Zeit brauchten. Diejenigen, die in der Verarbeitung schnell erschienen, waren offensichtlich kaum bedienbar. Es wurden nicht weniger als fünf Programme vorgestellt. Dank der benutzerfreundlichen GEM Oberfläche und der hohen Rechnergeschwindigkeit dürften die vorher genannten Probleme beim Atari nicht auftreten. Wenn auch die Produkte in diesem Jahr nicht mehr verfügbar sind, sahen die ersten Test-Versionen schon recht brauchbar aus.

Graphic Artist, das schon in Deutschland erhältlich ist (Vertrieb: KFC, Königstein), wurde um Bibliotheken zum Desktop Publishing ergänzt. Dieses Programm scheint jedoch eher ein CAD Paket als ein Programm zur grafikorientierten Textverarbeitung zu sein, obwohl die Demoausdrucke mehr als befriedigende Ergebnisse zeigten.

Boffin und **1st_Word Plus**, die teilweise als Desktop Publishing bezeichnet werden, machen als Textverarbeitung mit Grafik Option einen recht guten Eindruck, verfügen aber keinesfalls über Qualitäten, die diese Art von Programmen ausmachen.

Ganz anders dagegen das Programm **Publishing Partner** der SoftLogic Corp. (Vertrieb: Knupe, Dortmund), welches die Möglichkeiten des Apple Laserwriters und jeder Postscript-fähigen Einheit unterstützt. Ausgaben von 6 bis 144 Punkten Spiegelschrift. Invertieren und seitenverkehrtes Schreiben sind nur einige der Möglichkeiten dieses Paketes. Als wichtigstes bei diesem Paket erscheint mir die Ausgabe WYSIWYG – „What you see, is what you get“, d. h. man sieht alles auf dem Bildschirm so, wie es hinterher auf dem Drucker ausgegeben wird, auch die volle Druckerseite, natürlich auf dem Monitor verkleinert dargestellt. Dieses Produkt soll ab Ende Januar 1987 auch in Deutschland verfügbar sein, wie mir Hersteller und Distributor auf der Comdex erklärten.

LaserType von Softlab ist ein weiteres Programm dieser Art, welches bereits auf dem IBM PC verfügbar ist. Mit der Atari Version kann zum Frühjahr 1987 gerechnet werden. LaserType ist eine Textverarbeitung, die speziell für den HP-Laserjet+ unterstützt, ebenfalls unter GEM läuft, zusätzlich jedoch mit sehr vielen Tastatur-Zusatzbefehlen unterstützt werden muß.

Die Firma XLENT stellte ebenfalls ein Programm vor, welches in diese Kategorie einzuordnen ist. **Typesetter Elite** stellt eine radikale Weiterentwicklung des bekannten Typesetters dar. Zusätzlich ist das Programm **Mega Fonts** erschienen, um Zeichensätze selbst zu definieren. Beide Programme sind im Vertrieb von Computer Technik Kieckbusch, Ransbach und ebenfalls ab Januar erhältlich.

Ferner wurde, jedoch nicht auf dem Atari Stand, **Word Perfect** angekündigt.

Word Perfect ist ein Textverarbeitungsprogramm das auf IBM PC's bereits einen großen Erfolg hat. Dieses Paket wurde gleichzeitig für den Apple Macintosh und für den Commodore Amiga vorgestellt. Genaue Liefertermine und Vertriebswege waren leider nicht zu erfahren.

Atari selbst hat eine Textverarbeitung vorgestellt, die **Microsoft Word** sehr ähnlich ist. Ob dieses Produkt den „Standard" von First Word ablösen kann wird sich in der Zukunft erweisen.

Bei den Programmiersprachen waren alle namhaften Hersteller vertreten. Neue Programmiersprachen außer **Cambridge Lisp**, das seit ca. 6 Wochen verfügbar ist, wurden nicht gezeigt. Vertreten waren die Firmen Metacomco, Mark Williams und TDI. Wie mir Andrew Spencer von Metacomco mitteilte, arbeite man in England hart an einer neuen Version des **Lattice C** Compilers, der sicherlich inzwischen genau wie der Megamax C Compiler zum Standard in diesem Bereich geworden ist.

Auf meinem Messerundgang fiel mir in der Westhalle ein leistungsstarkes CAD Paket auf, das 1 : 1 vom IBM PC übernommen wurde. Der **Drafix** Source wurde auf dem ST neu compiliert und lief, bis auf geringfügige Änderungen in der Bildschirmausgabe. Drafix wird zunächst nur für den monochrom Monitor angeboten, weil ein CAD Paket dieses Umfangs nur mit dieser Auflösung arbeiten kann. Das komplette Paket soll ab Januar verfügbar sein und über Knupe, Dortmund vertrieben werden. Der Preis von unter DM 1000,– ist schon erstaunlich. Das CAD Paket wurde sowohl auf IBM AT-03 als auch auf dem Atari 1040 gezeigt. Interessant war, daß der Atari bei der Berechnung fast doppelt so schnell war wie der AT.

Als weitere CAD- oder Zeichenprogramme wurden noch Grafik Artist (bereits oben erwähnt) gezeigt und eine neue Version von **Easy Draw**. Ein Update soll für ca. DM 50,– von CTK, Ransbach für registrierte Endbenutzer zur Verfügung stehen.



Bild 2: Publishing Partner von Soft Logic

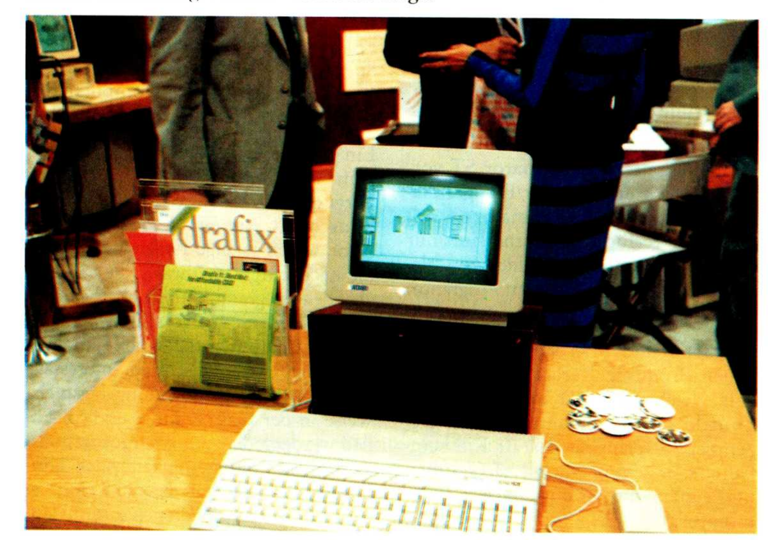


Bild 3: CAD auf dem ST mit Drafix

Degas Elite, von Tom Hudson, wurde ebenfalls gezeigt und ist bereits in Deutschland lieferbar. Es ist die konsequente Weiterentwicklung von Degas, das sich bei den Zeichenprogrammen bereits einen guten Namen geschaffen hat.

Auch bei den Tabellenkalkulationen hat sich etwas getan. **VIP Professional** ist in GEM Version gezeigt worden. Es ist direkt nach der Comdex in Stückzahlen lieferbar. Auch die deutsche Version soll verfügbar sein, wurde jedoch auf der Messe aus verständlichen Gründen (welcher Amerikaner kann schon deutsch?)



Bild 5: Der Berührungsbildschirm

nicht gezeigt. Festzustellen war, daß das neue VIP wesentlich schneller arbeitet als die alte Version. Angesichts dieser Tatsache ist die Preiserhöhung um rund 30 % schon zu verstehen. Wie und ob sich der erhöhte Preis auch in Deutschland niederschlagen wird, muß man abwarten. Eine weitere Tabellenkalkulation die voll unter GEM läuft, war ebenfalls zu sehen und ist ab Januar in Deutschland bei CTK und Knupe erhältlich.

Data Becker Produkte wurden auf dem Tisch des amerikanischen Softwarehauses Abacus gezeigt. Alle von Abacus gezeigten Programme fanden reges Interesse bei den Besuchern. Speziell das Programm **Platine ST** (so der deutsche Name) fand besondere Beachtung.

Das Angebot an Spielen wächst ständig. Der **Flugsimulator II** von Sublogic wurde in der Farbversion gezeigt. Dieser ist ab sofort lieferbar. Auf die Schwarz-Weiß Version müssen wir noch warten, da in USA hauptsächlich der Farbmonitor verkauft wird und Sublogic zunächst den US Markt sieht.

Aegis stellte seinen **Animator ST** vor, der wirklich erstaunliche Farbgrafikanimation zeigte. Der Animator ST soll ab Januar verfügbar sein und läuft nur auf dem Farbmonitor. Bilder von Neochrome können mit diesem Programm weiter verarbeitet und bewegt werden.

Michtron und Microdeal zeigten die neuen Produkte **Karate Kid II** (nach dem gleichnamigen Film), und **Space Shuttle**. Beide Programme sind ab Dezember 86 verfügbar. Karate Kid beinhaltet sehr schöne animierte Grafiken und läuft natürlich nur auf dem Farbmonitor.



Bild 4: Animator ST oder als die Neochrome-Bilder laufen lernten

Desweiteren wurden von Activision, FTL, Electronic Arts und vielen anderen namhaften Herstellern die neusten Spiele vorgestellt. Die Spiele werden bei Erscheinen von der ST Computer Redaktion vorgestellt werden.

Ein ganz neues Anwendungsgebiet mit dem Atari ST wurde von Video Touch vorgestellt und zwar ein **Berührungsbildschirm**. Auf einem Atari Farbmonitor waren farbige Felder dargestellt, die nach Fingerberührung die entsprechenden Programmpunkte abarbeiteten, bzw. ein Menü aus verschiedenen Gerichten zusammenstellen ließ und dazu gleich die Rechnung schrieb. Interessant könnte so eine Anwendung beispielsweise in der Gastronomie und SB Läden werden oder auch als Informationssystem auf Ausstellungen oder ähnlichem, wo der Benutzer, ohne eine Tastatur oder Maus bedienen zu müssen, bestimmte Eingaben am Bildschirm durchführen kann. Ein komplettes Entwicklungssytem bestehend aus Atari 1040, modifiziertem Farbmonitor, einer Harddisk (20 MB) und einem Bon Drucker von Star wurde für ca. $ 5000,- angeboten.

Auf der Comdex Herbst 1986 wurden sehr viele neue und interessante Produkte für den Atari ST vorgestellt. Die Qualität der Programme nimmt ständig zu, so daß man sagen kann, daß der Atari auch in der Business Welt seinen Platz unter den führenden Computer Systemen finden wird. In Deutschland wird die nächste Cebit in Hannover sicherlich auch das ein oder andere hier angekündigte Produkt zeigen oder bereits früher in der ST Computer besprochen werden.

Rainer Kleinhans

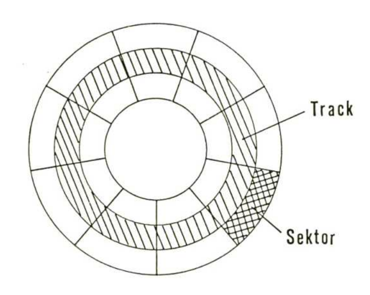
PASCAL ruft TOS

Teil 2 – Die Floppy

Auch diesem Speichermedium kann man in Pascal noch einiges entlocken, beispielsweise den direkten zugriff auf einzelne Sektoren oder auf das Directory. Die dafür verwendeten Routinen eignen sich zum Einbau in eigene Programm eund zeigen meist erst kombiniert ihre Leistungsfähigkeit.

Aufbau der Diskette

Zur besseren Verständlichkeit der folgenden Befehle wollen wir zunächst den Aufbau einer Diskette erläutern (siehe auch Grafik 1). Sie ist im unbenutzten Zustand noch unbrauchbar, denn erst das Formatieren der Diskette legt die Verteilung der einzelnen Bereiche fest. Das Standardformat einer ATARI ST-Diskette hat 80 Spuren, die in konzentrischen Kreisen um den Mittelpunkt angelegt sind. Jede Spur ist wiederum in neun Teilbereiche untergliedert, die sich Sektoren nennen und je 512 Byte umfassen. Mit diesen Sektoren arbeiten die nun folgenden Routinen.



Lesen und Schreiben eines Sektors

Die erste Routine dieser Folge ist der Direktzugriff auf bestimmte Sektoren der Diskette. Er ist im normalen Pascal-Wortschatz nicht enthalten, aber durch Einsatz der Betriebssystemroutinen erreichbar. Dafür gibt es z. B. im XBIOS des ST die passenden Routinen. Sie benötigen als Parameter folgende Werte:

- **Puffer** [n ★ 512 Bytes]: Adresse auf einen ausreichend großen Puffer, oder ein entsprechend dimensioniertes Feld
- **Füller:** unbenutzer Parameter
- **Drivenummer** (A=0, B=1)
- **Sektornummer** (0...9)
- **Tracknummer** (0...79) bzw. [0...82]
- **Diskettenseite** (0...1) je nach Laufwerk
- **Sektoranzahl** [n]: Anzahl der Sektoren, die hintereinander gelesen werden (die Zahl darf aber nicht größer sein als die Anzahl der Sektoren auf einem Track).

Der Aufruf zum Lesen lautet:

FLOPRD(buf,filler,devno,sectno, trackno,sideno,count)

Der Aufruf zum Schreiben lautet:

FLOPWR(buf,filler,devno,sectno, trackno,sideno,count)

Die Funktionen übergeben nach ihrer Ausführung einen **Statuswert**. Wenn der Wert Null ist, dann war der Aufruf erfolgreich, bei negativen Werten ist ein Fehler aufgetreten. Solche Werte sollten vom Programm abgefragt werden, um eventuell auftretende Fehler abfangen zu können.

Wenn man nun einen Sektor lesen oder schreiben will, beträgt der Puffer 512 Bytes und die Sektoranzahl '1'.

```
program DISC_COPY;

CONST disk_A=0;
      disk_B=1;
      vorne=0;
      hinten=1;

TYPE buff  = packed array [1..5000] of char;

VAR  dummy                               : long_integer;
     track,sector,count,drive,side,
     error,error1,error2,error3,error4   : integer;
     daten1,daten2                       : buff;
     ch                                  : char;

function FLOPRD (VAR buffer: buff; dummy: long_integer; drive,
            sector,track,side,count : integer ): integer; xbios(8);

function FLOPWR (VAR buffer: buff; dummy: long_integer; drive,
             sector,track,side,count : integer ): integer; xbios(9);


begin               { Hauptprogramm }

  sector:=1;
  count:= 9;
  writeln (chr(27),'E     Backup   A --> B');
  writeln;
  writeln (' Original-Diskette in Laufwerk A');
  writeln ('     Ziel-Diskette in Laufwerk B');
  writeln;
  writeln (' (q)uit');
  readln (ch);
  if ch<>'q' then begin
    for track:=0 to 79 do begin
      error1:=floprd (daten1,dummy,disk_A,sector,track,vorne,count);
      error2:=floprd (daten2,dummy,disk_A,sector,track,hinten,count); {SF 314
      error3:=flopwr (daten1,dummy,disk_B,sector,track,vorne,count);
      error4:=flopwr (daten2,dummy,disk_B,sector,track,hinten,count); {SF 314
      writeln(chr(27),'Y',chr(40),chr(37),'  Track ',track:3);
      writeln(' ******* Status *******');
      writeln(error1:5,error2:5,error3:5,error4:5);
    end;
    writeln ('   Backup ready  !!')
  end;
  repeat until keypress;

end.
```

Listing 1

Beim Beispiel des einfachen **Diskcopy-Programms** (Listing 1) werden aus Zeitgründen immer neun Sektoren, also ein gesamter Track, gelesen; der Puffer muß deshalb ausreichend dimensioniert werden. Die gelesenen Daten werden danach sofort auf die Diskette in Laufwerk B geschrieben. Das Programm benötigt deshalb zwei Laufwerke. Außerdem ist zu beachten, daß es momentan von zweiseitigen Laufwerken ausgeht, denn es wird jeweils ein Track auf der Vorder- und Rückseite gelesen. Bei Verwendung von einseitigen Laufwerken entfallen dann die Lese- und Schreibbefehle für die zweite Seite (mit {SF 314} gekennzeichnet).

Während des Kopierbetriebs werden die Nummern des gerade in Arbeit befindlichen Tracks und die Rückgabewerte angezeigt. Das Programm macht keine Fehlerabfrage, die Fehlerwerte werden jedoch sehr kurz (!) angezeigt. Eine Verbesserung an dieser Stelle ist dringend zu empfehlen.

Will man den Inhalt eines Sektors näher untersuchen oder anschauen, so ist die Definition des Puffers entscheidend. Ist dieses Feld als CHAR definiert, erhält man den Sektorinhalt als Zeichen, bei Definition als BYTE erscheinen die entsprechenden Zahlenwerte.

Laufwerkskennung

Manchmal ist es notwendig, das momentan aktuelle Laufwerk zu kennen. Dafür gibt es die Routine **DGETDRV**. Sie liefert den gewünschten numerischen Wert des momentanen Laufwerks und, nach einer einfachen Umrechnung, auch den Kennbuchstaben (siehe Listing 2).

Das Laufwerk, von dem ein Programm geladen wurde, ist das momentan aktive Laufwerk. Es wird beim Diskettenzugriff ohne Diskangabe angesprochen. Will man ein anderes Laufwerk zum aktiven erklären, verwendet man folgende Routine:

altdrive:= DSETDRV (neudrive)

wobei Laufwerk A=0; B=1; C=2; ...

In altdrive wird die Nummer des vorherigen aktiven Laufwerks zurückgegeben.

Speicherplatz

Über die Aufteilung eines Massenspeichers in Cluster, Sektoren, Byte, belegte und freie Bereiche gibt die Gemdos-Funktion **DFREE** Auskunft. Sie benötigt als Angabe nur die Laufwerksnummer (wobei Null dem aktuellen Laufwerk (!) entspricht, A=1, B=+ usw.) und gibt dann in einem Puffer vier Werte zurück:

buff [1] Anzahl der freien Cluster
buff[2] Anzahl der Cluster
buff [3] Anzahl der Byte/Sektor
buff[4] Anzahl der Sektoren/Cluster

Daraus läßt sich dann die Anzahl der freien, belegten und aller verfügbaren Bytes berechnen (siehe Listing 3).

Directory

Eine bei manchen Anwendungen wichtige Routine ist die Anzeige des Directories. Dazu sind mehrere Systemaufrufe nötig. Der erste lautet **FSETDTA** und setzt die Disketten-Transfer-Adresse, ab der **DTA-Puffer** steht. In diesem Puffer werden, von

```
program LAUFWERK;

VAR
 drive : char;

 function DGETDRV: integer; gemdos( $19 );

begin
   drive:=chr(DGETDRV+65);
   write('Sie benutzen momentan ');
   writeln('Laufwerk ',drive);
   readln
end.
```

Listing 2

```
program MEMORY;

TYPE buf4 = array [1..4] of long_integer;

VAR  buff      : buf4;
     drive,r   : integer;
     memory,freemem,usedmem   : long_integer;

function DFREE (VAR buff: buf4; drv: integer): integer;
                                               gemdos($36);

begin
   write('welches Laufwerk ( 0=akt., 1=A, 2=B, usw. ): ');
   readln(drive); writeln;
   r:=DFREE(buff,drive);    { 1 = Laufwerk A, 2=B, usw. }
   writeln('Funktionsrueckgabe        : ',r);
   writeln;
   writeln('freie Cluster             : ',buff[1]);
   writeln('Anzahl der Cluster        : ',buff[2]);
   writeln('Anzahl Byte/Sector        : ',buff[3]);
   writeln('Anzahl Sectoren/Cluster: ',buff[4]);
   writeln;
   freemem:=buff[1]*buff[3]*buff[4];
   writeln('frei   : ',freemem,' Byte');
   usedmem:=(buff[2]-buff[1])*buff[3]*buff[4];
   writeln('belegt : ',usedmem,' Byte');
   memory:=buff[2]*buff[3]*buff[4];
   writeln('gesamt : ',memory,' Byte');
   readln
end.
```

Listing 3

den folgenden Befehlen, die kompletten Directory-Informationen abgelegt. Der Puffer muß eine Größe von 44 Byte haben und kann als ARRAY oder RECORD angelegt werden. Er enthält folgende Informationen:

Byte	Inhalt
1...21	**reserviert** für TOS
22	**Attribut**
23...24	**Uhrzeit**
25...26	**Datum**
27...30	**Filelänge** (hexadezimal)
31...44	**Filename** und Extention

Nach Festlegung der Puffer-Adresse erfolgt der Aufruf **FSFIRST**. Dieser sucht das erste Diskettenfile, das mit dem angegebenen Muster (Pfadname) übereinstimmt und schreibt die Informationen in den Puffer. Dieses Muster kann auch 'Joker' (Bruchstücke eines Namens) enthalten (z. B. *.PAS oder A???????.PAS). Der zweite Parameter dieser Funktion ist das Dateiattribut, das eine weitere Selektierung der zu suchenden Dateien festlegen kann. Wenn hier eine Null übergeben wird, sind die Unterdirectories ausgeblendet, bei einem Wert von 16 werden auch sie angezeigt (siehe Tabelle).

Wert	Bedeutung
0	**Schreib-/Lesedatei**
1	**Nur-Lesedatei**
2	**versteckte Datei**
4	**System-Datei**
8	**Volume-Label**
16	**Unterdirectory**

Zum Suchen eines weiteren Eintrages dient die Funktion **FSNEXT**. Diese benöigt keine Parameter (die von FSFIRST gesetzten sind weiterhin gültig), liefert aber eine Fehlernummer und schreibt die Fileinformation in den DTA.

Das Programm **DIRECTORY** (Listing 4) zeigt, wie das Directory eines beliebigen Laufwerkes gelesen und auf dem Bildschirm angezeigt wird. Dieses Programm verwendet fast alle der bereits besprochenen Routinen und gibt die Directory in 'aufbereiteter' Form aus (siehe Bild 2). Das jeweilige Directory-File wird hier in einem Record abgelegt, um einen unkomplizierten Zugriff auf die einzelnen Elemente zu haben. Bei näherer Betrachtung dieses Records wird Ihnen auffallen, daß das Datum und die Uhrzeit in jeweils einer Integer-Zahl abgelegt werden. Der genaue Aufbau beider Zahlen

```
 Datum         Zeit        Byte  Name
-----------------------------------------
20 11 1985     0: 4      117062  BIRD2.ANI
20 11 1985     0: 5       32128  BIRD2.NEO
20 11 1985     0: 5       32128  HOUSE3.NEO
20 11 1985     0:18      117062  AAFALL.ANI
20 11 1985     0: 6       32128  TINAFIN.NEO
20 11 1985     0: 6       11493  BOINK.PRG
20 11 1985     0: 6       33634  NEO.PRG
20 11 1985     0: 6        5873  SLIDEANI.PRG
20 11 1985     0:37       53517  COPYBOX.S
20 11 1985     0: 6       32128  AAFALL.NEO
20 11 1985     0: 6       32128  AAINSECT.NEO
20 11 1985     0: 6       32128  AATRAIN.NEO
20 11 1985     0: 7       32128  CHROME.NEO
20 11 1985     0: 7       32128  FRACTAL.NEO
20 11 1985     0: 7       32128  GREATWAV.NEO
20 11 1985     0: 8       32128  ROBOTTV.NEO
20 11 1985     0: 8       32128  SCICOVER.NEO

frei    : 0 Byte
belegt  : 728064 Byte
```

Bild 2

```
program DIRECTORY;

  TYPE
    nametyp   = packed array [1..14] of char;
    path_name = packed array [1..80] of char;
    DIRREC = record
               reserved : packed array [0..21] of byte;
               time     : integer;
               date     : integer;
               size     : long_integer;
               filename : nametyp;
             end;

  VAR
    dirfile : dirrec;
    wdh  : integer;
    name : string;
    path : path_name;
    ch : char;

  function DGETDRV: integer; gemdos( $19 );

  procedure FSETDTA( VAR buf : dirrec ); gemdos( $1a );

  function FSFIRST( VAR path: path_name; search_attrib: integer ):
                                                integer; gemdos( $4e );

  function FSNEXT : integer; gemdos( $4f ) ;

  procedure SET_DRIVE;
    VAR r,drive : integer;
        drv     : char;

    function DSETDRV( drive: integer ): integer; gemdos( $0E );

    begin
      write('bitte Laufwerk angeben (A, B, C, usw.): ');
      read(drv);
      if drv in ['A'..'Z'] then drive:=ord(drv)-65
                           else drive:=ord(drv)-97;
      r:=DSETDRV(drive);
      if r<0 then begin
          writeln('Fehler bei der Eingabe !');
          readln
      end
    end;
```

```
procedure SHOWFILE( VAR dirfile : dirrec ) ;
  VAR i,jahr,monat,tag,stunden,minuten : integer;

  begin
    with dirfile do begin
      jahr:=shr(date,9);
      monat:=shr((date-shl(jahr,9)),5);
      tag:=date-shl(monat,5)-shl(jahr,9);
      write( tag:2,' ',monat:2,' ',jahr+1980:4,'  ' );
      stunden:=shr(time,11);
      minuten:=shr((time-shl(stunden,11)),5);
      write( stunden:2,':',minuten:2,'  ' );
      write( size:8,'  ' );
      i := 1 ;
      while filename[i]<>chr(0) do begin
        write( filename[i] ) ;
        i := i + 1
      end;
      writeln
    end
  end;

procedure MEMORY;

  TYPE buf4 = array [1..4] of long_integer;

  VAR  buff       : buf4;
       freemem,usedmem  : long_integer;

  procedure DFREE (VAR buff: buf4; drv: integer); gemdos( $36 );

  begin
    writeln;
    DFREE(buff,0);        { 0 fuer aktuelles Laufwerk }
    freemem:=buff[1]*buff[3]*buff[4];
    writeln('frei   : ',freemem,' Byte');
    usedmem:=(buff[2]-buff[1])*buff[3]*buff[4];
    writeln('belegt : ',usedmem,' Byte');
  end;

procedure PFAD_NAME;
 VAR i : integer;
 begin
    write( 'Pfadname: ' );             { Pfad }
    readln( name );
    if name='' then begin
      name:='*.*';
      writeln( name )
    end;
    name:=concat( name,chr(0) );
    for i := 1 to length( name ) do path[i] := name[i] ;
 end;

procedure DIRECTORY_LESEN;
 begin

    FSETDTA( dirfile ) ;                         { setzt DTA }
    if FSFIRST( path, 16 ) >= 0 then begin       { erster Eintrag }
      writeln(' Datum       Zeit      Byte  Name');
      writeln('-----------------------------------------');
      repeat
        SHOWFILE( dirfile );                    { Anzeigen }
        wdh:=FSNEXT;                      { weitere Eintraege }
      until wdh < 0;
      MEMORY;
    end
    else
        write( ' keine Datei gefunden ! ' );
    readln;
end;

begin       { Hauptprogramm }

  repeat
    writeln(chr(27),'E');
    writeln('MENUE');
    writeln('                    aktuelles Laufwerk:',chr(DGETDRV+65));
    writeln('(D)irectory lesen');
    writeln('(A)usdrucken');
    writeln('(L)aufwerk aendern');
    writeln('(Q)uit');
    writeln; write('Eingabe: ');
    read(ch); writeln;

    case ch of
      'd','D' : begin
                  PFAD_NAME;
                  DIRECTORY_LESEN;
                end;
      'a','A' : begin
                  PFAD_NAME;
                  rewrite (output,'PRN:');
                  DIRECTORY_LESEN;
                  rewrite (output,'CON:')
                end;
      'l','L' : SET_DRIVE;
    end;
 until ch='q'

end.
```

Listing 4

DATUM (Format: INTEGER)

Bit	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
	Tag					Monat				Jahr+1980						

UHRZEIT (Format: INTEGER)

Bit	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
	Sekunden'					Minuten						Stunden				

(' Sekunden in Zweierschritten)

Grafik 3: Darstellung von DATUM und ZEIT

kann der Grafik 3 entnommen werden. Die einzelnen Daten (z. B. Tag, Monat, Jahr) müssen erst voneinander 'getrennt' werden. Diese Arbeit erledigt das Programm mit einigen Schiebe-Befehlen **(ShR & ShL)**, die man sich eventuell etwas genauer anschauen muß.

Der eigentliche Filename ist in einem ARRAY of CHAR abgelegt und wird als Einzelbuchstabe ausgegeben, bis sein letztes Zeichen **(chr(0))** erreicht wird.

In den nächsten Ausgaben folgen Tips zu

- Filebehandlung (Schützen, Löschen, Umbenennen, ...)
- Sound (Erzeugen und Abspielen von Musik) (Interessante Nebeneffekte)
- I/O (Die Schnittstellen: RS232, parallel, MIDI, Tastatur)

(MN & HS)

GFA-BASIC
Ein Basic, auf das Sie schon lange gewartet haben. Wir erleben täglich helle Begeisterung.
- kompakter Hochgeschwindigkeits-Interpreter.
- 11-stellige Genauigkeit
- Strukturiertes Programmieren.
- Einfachste GEM-Programmierung.
- Komfortabler Editor.

Ein absolutes "Muß" für jeden Atari-Besitzer. Testbericht siehe ST-Computer Nr. 7 Juli/August
GFA-Basic **149,-** Buch zum GFA **49,-**
Programm Disk **39,-**

WIR SIND IHR STARKER
ATARI
PARTNER

Megamax C
Ein komplettes C-Entwicklungssystem
- Single Pass-Compiler
- Inline Assembler
- Disassembler
- Linker & Librarian
- Resource Construktion
- Vollständige GEM-Libraries
- Unix-Routinen
- GEM-Editor, GEM-Shell
- 370 Seiten-Handbuch

DM 495,-

Händleranfragen erwünscht

Adress-access 2
Die phantastische Adressverwaltung für den Atari ST
- GEM-Funktion
- Menü-Wahl mit Maus oder Funktionstasten
- Gleichzeitig Suchen nach allen Textfeldern
- Sortieren nach beliebigem Feld
- Ausgabe auf Etiketten und Formularen
- Bildschirmausgaben mit vertikalen und horizontalem Scrolling
- Mit deutschem Handbuch für alle Atari ST's **99,-**

Wordstar/Mailmerge
Der Bestseller unter den Textverarbeitungsprogrammen bietet Ihnen bildschirmorientierte Formatierung, deutscher Zeichensatz und DIN-Tastatur sowie integrierte Hilfstexte
- Mailmerge für Serienbriefe
- Beliebige Anzahl Adressen
- Drucken von Adressaufklebern
- Für beliebige Drucker (centronics)

Programm: **199,-**
Buch zum Programm: **49,-**

G-Data Software
Ausgesuchte Programme mit hohem Gebrauchswert.

G-Harddisk-Help: Sicherheitsduplikat auf Disketten **79,-**

G-Utilities: G-Format, Time, Free, List, Trans, Verify Datamax Prim, RCS Superbackup **65,-**

G-Diskmon: Alles um den Sektor String und Bytes **89,-**

G-Ramdisk: Wird als Desk Accessory bei jedem Systemstart mitgebootet. 100mal schneller als Laufwerke. Nach Reset und neuem boot bleibt Inhalt erhalten. **48,-**

as-adress: Adressverwaltung in komfortabler GEM-Umgebung. 1st-word kompatible Harddiskhandlung integriert (backup) 45 000 Adressen auf Harddisk 1800 pro 720 K Disk. **95,-**

Omikron-Basic
- Extrem schnell - Lieferb. auf Modul
- Prozeduren mit Übergaben und Rückgabenparam. u. lok. Variablen
- 6 Variablentypen
- Im mathemat. Bereich unschlagbar: 19 stellige Rechengenauigkeit 54 mathemat. Funktionen u. sehr schnelle Opertoren
- Unterstützt prof. kaufm. Programmierung, z. B. Masken-Input, Sortierbefehle, Isam-Dateiverwaltung
- VDI- u. AES-Funkt. direkt über GEM-Library aufrufbar
- Echter Direktmod. mit voll. Screen-Editius **229,-**

Boffin
- Neueste Textverarb. unter GEM (15 Minuten Lernzeit)
- Mischt Grafik + Text
- Versch. Schriftarten u. -größen möglich
- Zeichnet Geschäftsgrafiken (Balken-, Liniendiagramme etc.)
- Beherrscht die Grafik

Farbige Grafiken und Zeichnungen aus Degas, Doodle, Neochrom und Metafiles werden eingelesen und fürs Zeichnen während des Textes kann auch das angebotene Zeichenprogr. Kreis, Linie, etc. über ein Pull-Down-Menü aufgerufen werden. **398,-**

Mono-Star
- Pixelorientiertes Programm, also kreativ
- Funktion nur im schw./w.-Modus
- kpl. in GFA-Basic geschrieben
- Wenig benutzte Funktionen sind herausgelassen oder durch neue ersetzt (z. B. Biegen u. Verzerren von Bildteilen, fließende Verkleinerungen und Vergrößerungen, automatisches Glätten von Freihandzeichnungen
- Mono-Star, das absolute Muß für kreativ und künstlerisch arbeitende ST-Anwender **99,-**

Color-Star
- wie "Mono Star", aber Farbe
- Farbanimation mit 512 Farben
- GEM unterstützt
- verformen von Zeichnungen
- mischen von Farben (ca. 16 000 Farben)
- Das Grafikprogramm für Ihren ST. Werden Sie kreativ!

99,-

Cumana
Laufwerke
3,5 Zoll Doppellaufwerk 80 Spuren doppelseitig 1 MB, Anschlußkabel und Netzkabel 699,-

3,5 Zoll Doppellaufwerk 2 x 80 Spuren doppelseitig 2 x 1 MB, Anschlußkabel, und Netzkabel (nicht für Atari 1040 ST) (auch übereinander) **1099,-**

5 1/4 Zoll Slimline-Laufwerk 40/80 Spuren umschaltbar doppelseitig 1 MB komplett mit Netzkabel und Anschlußkabel **759,-**

5 1/4 Zoll Slimline Doppellaufwerk 2 x 40/80 Spuren umschaltbar 2 x 1 MB komplett mit Netzkabel und Anschlußkabel (nicht für Atari 1040 ST) **1299,-**

«mixed« slimline Doppelstation mit einem 3,5 Zoll und einem 5 1/4 Zoll Laufwerk 40/80 Spuren umschaltbar 2 x 1 MB **1199,-**

Atari-kompatible Maus **120,-**

Cumana Laufwerke sind fast unhörbar, lang und kritisch getestet, einfach absolute Spitzenklasse!

Spiele:

1.	Borrowed Time	89,90
2.	Brataccas	89,90
3.	Colourspace	98,-
4.	Delta Patrol	129,-
5.	Dragonworld	149,-
6.	Flightsimulator SUB	178,-
7.	Hichthikers	99,-
8.	Land of Havoe	79,90
9.	Little Computer People	89,90
10.	Major Motion	79,90
11.	Quiwi	69,-
12.	Sun Dog	98,-
13.	Time Bandit	99,-
14.	The Hacker	89,-
15.	The Pawn	99,-
16.	The Black Couldron	99,-
17.	The Music Studio	99,-
18.	Top Secret	89,-
19.	Wintergames	98,-
	außerdem:	
20.	Arena	99,-
21.	Ballyhoo	89,-
22.	Deep Space	139,-
23.	Leader Board	89,-
24.	Starcross	99,-
25.	Silent Service	99,-
26.	The Witness	89,-
27.	Hanse	79,-

Kuma-Programme

K-Seka (Assembler)	**148,-**
K-Spread (Kalkulation, deutsch)	**148,-**
K-Graph (Grafik, deutsch)	**118,-**
K-Comm (deutsch)	**148,-**
K-Resource (Utilities)	**118,-**
K-Word (Text)	**118,-**
K-Ram (Ram-Disk)	**89,-**
K-Switch (Utilities)	**89,-**
K-Minstrel (Musik)	**99,-**

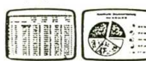
1st Word Plus
Erhebliche Leistungssteigerung in der Textbe- und -verarbeitung und erlaubt Grafikeinbindung in den Text.
- Briefe mit Bildern im Text
- Dokumentation mit Schaubildern oder dig. Fotos im Text
- Sourcecode-Erstell. für Programmiersprachen
- Volle GEM-Unterstützung
- Integration von Text und Grafik
- Rechtschreibprüfung (engl.)
- Textformattrennung / Dezimaltab.
- Komfortabler Suchen u. Ersetzen
- 1st Mail **99,-**
- Daten-Kompat. m. 1st-Word **298,-**

- Bestellen Sie per Telefon.
- Der Versand erfolgt am Bestelltag.

Potsdamer Ring 10 · 7150 Backnang · ☎ 07191/1528-29

weeske
COMPUTER-ELEKTRONIK

VIP-Professional
(Lotus 1, 2, 3)



1. Kalkulation
2. Datenbank **je DM 648,-**
3. Grafik

Der Traum für Manager und Selbständige.

WRITE 90°
- dreht Ihren Ausdruck um 90° (für überbreite Ausdrucke geeignet) **79,-**

TYPESETTER ELITE
Machen Sie Ihr eigenes Seitenlayout! - bindet Text und Grafik mit ein - druckt mit 960x672 Pixeln - liest fremde Fonts **139,-**

MEGAFONT ST
- liest Ihr 1st Word Text File (und andere) und druckt es in verschiedenen Fonts - Grafikeinbindung möglich - eigene Fonts erstellbar **89,-**

Graphik

1.	Animator	119,-
2.	Maps and Legends	198,-
3.	Degas	169,-
4.	Easy-Draw	398,-
5.	Mica	298,-
6.	ST-Draw	448,-
7.	ST-Colouring (Neochrombilder)	125,-
8.	Platine ST	698,-
9.	Profi Painter	99,-
10.	Paintworks	98,-
11.	Grafic Artist	698,-

Drucker

Star SG10	698,-
NL 10 mit Interface	898,-
Panasonic 1080	798,-
Panasonic 1091	998,-
Panasonic 1092	1098,-
Okidata ML 182	898,-
Okidata ML 192	1398,-
C-Itoh Riteman F+	898,-
Epson LX80	998,-
Epson FX85	1278,-
Epson FX86	898,-
Okimate 20	848,-
SMM 804	698,-

TELEFONISCHE BESTELLUNG: 07191 1528/29

Anwendungen

*Fußnotenverwalt. zu 1st Word	79,-
1-st-Word / 99,- deutsch	149,-
SM-Text (Sonderpreis)	180,-
Textomat ST	99,-
Datamat ST	99,-
DB-Master one	99,-
Text-Design ST	99,-
Profimat ST	99,-
DB-Man (Datenbank)	385,-
Trimbase (Datenbank)	298,-
Megabase	498,-
Format Copy V.1,4	89,-
Side-Click (Uhr, Termin Rechner)	145,-
BS-Handel (Bericht ST-7, Juli/August)	948,-
VT100 Terminal Emulator	195,-

Literatur

1. Für Einsteiger
2. Tips und Tricks
3. Peeks und Pokes
4. von Basic zu C
5. Das CP/M Buch
6. 3D Grafik und Programm
7. GEM Draw, Degas, Neochr.
8. Die C-Programmierung unter Tos
9. Atari ST Logo-Hndbuch

Neueste Spiele:

1. Shanghai	95,-
2. Baslutball	95,-
3. Juwels of Darkness	89,-
4. Tee up (Golf)	59,-
5. Super Huey	
6. Electronic Pool	79,-
7. The Pinball	79,-
8. Strip Poker	79,-
9. Q-Ball	79,-
10. Time Blast	59,-
11. Tass Times	95,-
12. Space Station	95,-
13. Karate	95,-
14. Protector	95,-
15. A Mind Forever Voaging	89,-
16. Cards	79,-
17. Infidel	89,-
18. Suspect	89,-
19. Sidewinder	89,-
20. Mercenary	89,-
21. Leather Goddesses	89,-

PUBLIC-DOMAIN-SOFTWARE

Wir liefern auf erstklassigem Diskettenmaterial die Public-Domain-Software 1-30 aus ST-Computer für **12,- DM je Diskette**. Siehe PD-Service dieser Ausgabe.

Ebenfalls erhältlich: Sämtliche in ST-Computer veröffentlichte Programme:

Jan-Feb	28,-
März-Apr	28,-
Mai-Juni	28,-
Jul-Aug	18,-
Sept-Okt	28,-
Nov-Dez	28,-

Außerdem erhalten Sie bei uns weitere PD-Software ebenfalls für 12,- DM je Diskette. Als Beispiel:

100. Diskettenverwaltungsprogramm
102. Das neue TOS
106. Convertierprg. von f auf s/w
108. Kermit.GEM Terminalprogramm
112. AD68 Assembler
114 Copyram.DSK Ramdisk mit eigenem Kopierprg.
119. Mike Oldfields FOREIGN AFFAIR digitalisiert
120. Diskette voller Grafikprogramme
127. Formatierprg. für 0-79 Tracks sowie 80-85
129. Erstellen von eigenen Fonts

Weitere sind in Vorbereitung! Ausführlicher Ktalog wird bei einer Bestellung mitgeliefert!

Wir suchen:
ständig gute Public-Domain-Software, die wir in unseren PD-Katalog mit aufnehmen können. Bieten Sie uns brauchbare Programme an!

Metacomco Software

Hochwertige Entwicklungspakete zum günstigen Preis!

Lattice C, der Compiler	**348,-**
Makro-Assembler	**168,-**
Pascal-Compiler	**248,-**
NEU: (Beschreibung in ST 11/86)	
Make	**168,-**
Lisp	**498,-**
BCPL-Compiler	**329,-**

Der Mac Emulator

Die Chance für Sie, auf Ihrem Atari ST Macintosh Software zu fahren.
- läuft dadurch 20 % schneller als auf dem Mac
- nutzt Ihren Monochrombildschirm voll aus (größer als Macscreen!)
- größere Speicherkapazität als Mac (bei 520 ST+, oder 1040 ST)

Steigen Sie mit dem Mac Emulator in die Welt des Macintosh ein!

Wichtiges Zubehör

Diskettenbox 3.5"/40	39,80
Diskettenbox 3.5"/80	46,80
Weide Erweiterung auf 1 MB (ohne Löten)	275,-
Eprommer Atari ST	348,-
Eprom-Karte	98,-
Steckplatzerweiterung	189,-
Pal-Interface (HF-Modulator)	298,-
Digitizer (digitale Graphik)	598,-
Disketten 1DD 10 Stück	49,-
Disketten 2DD 10 Stück	59,-
Schaltpläne für alle Atari	29,80
Druckerkabel	38,-
Druckerständer Rauchglas	98,-
Endlospapier 1000 Blatt	29,-
Monitor Dreh-Kipp-Fuß	48,-
Thomson-Farbmonitor 36512	898,-
Thomson-Farbmonitor 36382	1298,-
Orion-Farbmonitor	848,-
Einzelblatteinzug für NL 10	290,-
Farbband SG 10	9,80
Farbband NL 10, Oki	24,80
Farbband Epson, SMM 804	22,80
Farbband Oki 20, Riteman	19,80
Atari Trackball	98,-
CSF-Gehäuse für 260/520 ST	198,-
ROM-Satz für alle ST's	140,-
Div. Kabel u. Stecker a. A.	

Schaltplanservice

Für folgende Atari-Geräte:

Atari 260 ST / 520 ST	29,80
Atari 520 ST+	29,80
Atari 1040 STF	29,80
Atari 520 STM	29,80
Atari 314 SF	14,80
Atari 354 SF	14,80
Atari SNM 804	14,80
Atari 600	14,80
Atari 800	14,80
Atari 1050	14,80

•NEUHEITEN•NEUHEITEN•

Neue M+T-Programme

Protext ST

Eine professionelle Textverarbeitung für den ST

- Vollautomatische Silbentrennung mit einstellbarem Trenngrad
- Blockoperationen, Textattribute
- Druckbild auf dem Bildschirm
- Maus-unterstützt
- Schnelles Position-Suchen

DM 148,-

Speichererweiterung

Noch nie war es so einfach, seinen Computer aufzurüsten, wie es jetzt durch die

Weide-Speichererweiterung

möglich ist. Müheloser Einbau nur durch Stecken auf 512 K.

DM 275,-

Neue M+T-Programme

d'base II ST

Nun gibt es endlich den Standard unter den Datenbanksprachen auch für den Atari ST. Es unterstützt die Schnelligkeit des 68000-Prozessors voll und ganz und läßt sich komfortabel über Pull-Down-Menüs mit der Maus steuern.

Programm **DM 348,-**
Buch zum Programm **DM 49,-**

Echtzeituhr für Atari ST

Einbau der Uhr durch Einstecken in den Rechner, ohne Löten, sehr einfach.

- Datum, Uhrzeit
- Uhrenbaustein ist rtc 58321 h
- Langzeit-Genauigkeit
- Lithium-Batterien, 10 Jahre
- Einbau problemlos
- Programm für Auto-Ordner
- Schaltjahrerkennung

nur DM 129,-

Copy Star V1.8

Das Programm, das jeden Kopierschutz ignoriert und somit erlaubt, von Ihrem geschützten Programm eine Sicherheitskopie zu machen. Sie erhalten eine Normalkopie in 34 sec.!!!

Auch die Konvertierung in ein schnelleres Spezialformat und eine Erweiterung der Diskettenkapazität um 50 bzw. 100 KB ist damit möglich.

DM 169,-

Programmiersprachen

Die Entwicklungspakete

GST-Makro-Assembler	149,-
GST-C-Compiler	295,-
MCC-Makro-Assembler	168,-
MCC-Pascal-Compiler	248,-
MCC-Lattice C	348,-
Fast-Basic-Compiler	348,-
Fast-C-Compiler	398,-
Fast-Fortran 77	789,-
Fast-Pascal-Compiler	490,-
Prospero-Fortran 77	488,-
Prospero-Pascal	448,-
Laser Base	298,-
UCSD-P System Pascal	349,-
Modula 2 ST Compiler	349,-
Basic M-Compiler	398,-
Lisp-Compiler/Interpreter	490,-
4x Forth Level 1 (Superschnell)	348,-
4x Forth Level 2 (Mit Gem)	548,-
Forth ST	99,-
K-Seka Assembler	148,-

Atari-Trackball

Der Trackball als Alternative zur Maus, geringer Platzbedarf, gute Positionierung, mit Umbauanleitung **DM 98,-**
Anschlußfertig umgebaut **DM 128,-**

Marconi-Trackball

Höchste Präzision!!
Wurde in England für den Militärbereich entwickelt (Zielflugeinrichtung, Raketensteuerung, Panzerabwehr), bestens geeignet für CAM/CAD. Profigerät für Profianwender.
Anschlußfertig **DM 198,-**

Modula 2 ST

Modula 2 ist die Supersprache, die Ihnen alle Möglichkeiten von Pascal bietet und als besondere Feature das modulare Konzept, das besonders dem kommerziellen Programmierer die Arbeit wesentlich erleichtert.

- Volle GEM-Unterstützung
- Schnelles Laufverhalten
- Gute Dokumentation
- Sehr gute Ram-Disc wird mitgeliefert

DM 349,-

Katalog 2,- DM
(Schutzgebühr in Briefmarken)

Hiermit bestelle ich per Nachnahme:

____

____

____
Vorname, Name

____
Straße, Hausnummer

____
PLZ, Ort

____
Telefon-Nummer

____
Datum, Unterschrift

Gemkurs

Teil 2

von Th. Weinstein

Hier nun der zweite Teil von "GEM unter C". Ich hoffe, Sie haben den ersten Teil gut verdaut, so daß ich auf das dort angelegte Grundwissen aufbauen kann.

Mit Erscheinen dieses Teils sollte es wohl möglich sein, den gesamten Quellcode – sowohl von **FULLPLOT**, als auch von **BROWSER** – über den Verlag zu erhalten.

Bevor es losgeht, noch ein paar Worte in eigener Sache: Die in dieser Artikelserie vorgestellten Programme sind alle bereits von einem erheblichen Umfang. Es ist klar, daß mit der Länge die Fehlermöglichkeiten gewaltig steigen.

Bei „normaler" Benutzung sollte eigentlich nichts passieren. Sollte Ihnen trotzdem etwas auffallen, wäre ich Ihnen für eine entsprechende Mitteilung an mich oder die Redaktion sehr verbunden.

1. Die Idee

Auf fast jedem Rechner – und das sind einige, – an dem ich bis heute gearbeitet habe, habe ich ein Programm geschrieben, das es ermöglichte, sich die trockene Gestalt einer Funktion y = f(x) als mehr oder minder hübsche Kurve vom Computer zeichnen zu lassen.

Die erste Maschine, auf der ich das tat, war übrigens ein Großrechner der Firma Burroughs. Das Programm wurde über Lochkarten eingelesen und die Kurve kam als Ausdruck, bestehend aus Sternchen und Leerzeichen, aus einem 132 spaltigen Schnelldrucker. Das ist jetzt etwa 8 Jahre her, und ich erzähle Ihnen das, um zu zeigen, was für Fortschritte die „Computerei" doch inzwischen gemacht hat. Wer hätte es sich damals träumen lassen, daß es einmal Rechner wie den ATARI ST geben würde, die man sich, zu einem erschwinglichen Preis, daheim auf den Schreibtisch stellen kann.

Mit den gewachsenen Ansprüchen der letzten acht Jahre im Rücken stellte ich mir folgenden Katalog an Fähigkeiten zusammen, die das Programm haben sollte:

1.) Es sollte auf jeden Fall möglich sein, die Funktion direkt ins laufende Programm einzugeben. Wenn man bei jeder Änderung alles neu übersetzen muß, wird das ganze schnell langweilig.

2.) Es sollte nicht nur die Visualisierung der Funktionsgleichung, sondern auch ihrer Ableitung und (einer) Integralkurve möglich sein.

3.) Ich wollte mehrere Funktionen gleichzeitig sehen können, um z. B. Funktion und Ableitung gegenüberstellen oder verschiedene Ausschnitte aus einer Kurve gleichzeitig sehen zu können.

4.) Es sollten verschiedene Methoden der Skalierung vorhanden sein.

5.) Um die Sache nicht zu sehr zu komplizieren, sollte nur der rein qualitative Verlauf einer Kurve gezeigt werden, also keine Achsenbeschriftung oder ähnliches.

Es ist vielleicht nicht so offensichtlich, aber der Punkt 1 ist der am schwierigsten zu realisierende am ganzen Programm.
Von Interpretersprachen aus, wie z. B. BASIC, ist es relativ simpel, während des Programmlaufs sich beliebig ändernde arithmetische Ausdrücke einzugeben, da das Programm ja nicht in übersetzter, sondern in einer mehr oder weniger textuellen Form vorliegt. Man kann also dem Interpreter den Text meist irgendwie der neuen Funktion unterschieben und ihm die Arbeit der Auswertung überlassen.

Ganz anders sieht die Angelegenheit in Compilersprachen wie 'C' aus. Wenn das Programm übersetzt ist, liegt es in der Sprache des jeweiligen Prozessors vor und ist eigentlich nicht mehr veränderbar. zur Lösung dieses Problems später noch ein paar Bemerkungen.

Punkt 2 ist relativ einfach, wenn man auf große Genauigkeit keinen Wert legt.

Punkt 3 schreit geradezu nach Fenstern

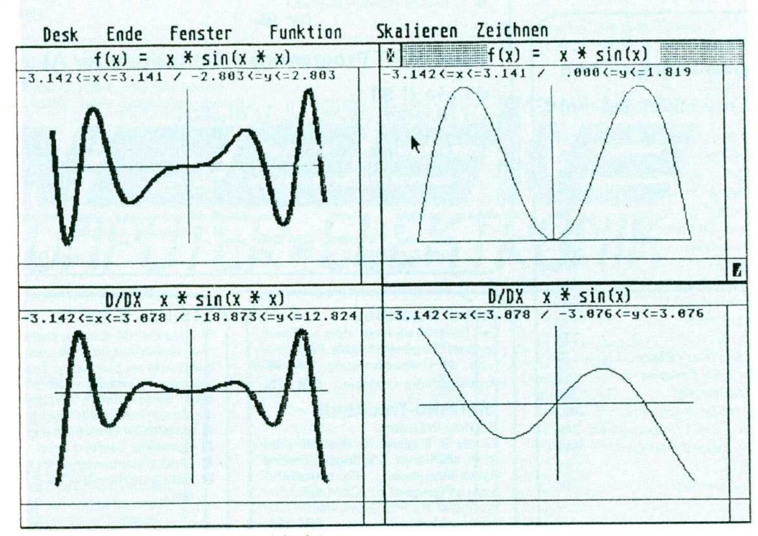


Bild 1: Hardcopy vom Bildschirm

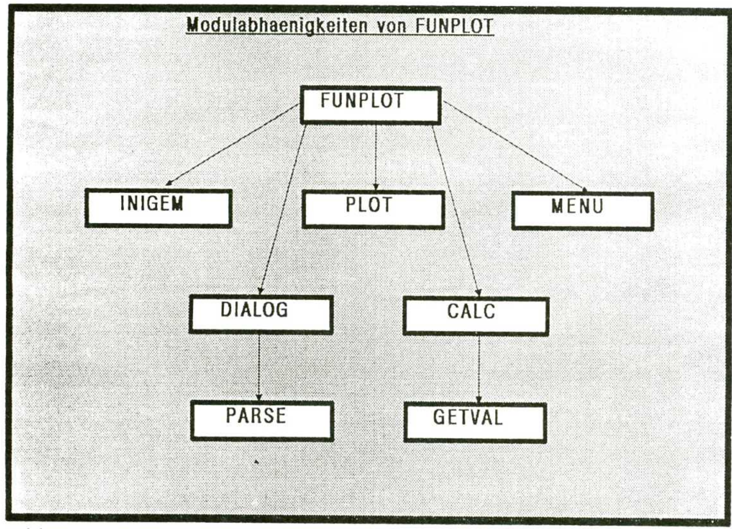


Bild 2: Die Module von FUNPLOT

```
/*
 * PARSE.C
 *
 * Version 1.0 vom 30.08.86
 * von Thomas Weinstein
 * geschrieben unter MEGAMAX C Entwicklungssystem
 *
 * Realisiert einen Parser fuer arithmetische Ausdruecke der Form
 *
 *                y = f(x,a,b,c,d)
 *
 * wobei a,b,c,d Konstanten sind und x die unabhaengige Veraenderliche
 *
 * Der Zerteiler arbeitet mit folgender Grammatik:
 *
 *       ZEILE          ::=    AUSDRUCK EOL.
 *       AUSDRUCK       ::=    TERM MEHR_TERME.
 *
 *       MEHR_TERME     ::=    + TERM MEHR_TERME |
 *                             - TERM MEHR_TERME |
 *                             EPSILON.
 *
 *       TERM           ::=    SFAKTOR MEHR_FAKTOREN.
 *
 *       MEHR_FAKTOREN ::=     * SFAKTOR MEHR_FAKTOREN |
 *                             / SFAKTOR MEHR_FAKTOREN |
 *                             EPSILON.
 *
 *       SFAKTOR        ::=    + FAKTOR |
 *                             - FAKTOR |
 *                               FAKTOR.
 *
 *       FAKTOR         ::=    BEZEICHNER    |
 *                             ZAHL          |
 *                             (AUSDRUCK)    |
 *                             FUN(AUSDRUCK).
 *
 * Die folgenden Produktionen werden von 'lexscan' verarbeitet:
 *
 *       BEZEICHNER  ::=    'X' | 'A' | 'B' | 'C' | 'D'.
 *
 *       ZAHL        ::=    GANZZAHL | '.'GANZZAHL | GANZZAHL'.'GANZZAHL.
 *
 *       GANZZAHL    ::=    ZIFFER | ZIFFER GANZZAHL.
 *
 *       FUN         ::=    'SIN' | 'COS' | 'TAN' | 'ATAN' | 'EXP' |
 *                          'LOG' | 'SQRT'| 'INT' | 'ABS'  .
 */

0
```

Listing 4

und wurde in **FUNPLOT** auch so verwirklicht. Da einer Applikation nur 4 Fenster zur Verfügung stehen, mußte ich mich auf diese Zahl beschränken. Es ist also möglich, maximal 4 verschiedene Funktionen gleichzeitig zu betrachten.

Auf die restlichen Punkte werde ich bei der Besprechung des Programms näher eingehen.

Was aus den obigen Überlegungen am Ende geworden ist, können Sie in Bild 1 sehen. Ich habe noch ein paar Spielereien eingebaut, so kann man etwa bei der Darstellung zwischen dünnen und dicken Linien wählen.

Die Benutzung wird sich normalerweise so gestalten:

- Man gibt unter Punkt **Funktion** eine neue Funktion ein und wählt das darzustellende X-Intervall.
- Unter **Skalieren** wählt man die gewünschte Skalierungsart. Es stehen zur Verfügung:

  Y gleich X:
  Der Y Bereich wir dem X-Bereich angepaßt.

  Y-Intervall:
  Der Benutzer kann ein beliebiges Intervall angeben, in dem die Werte dargestellt werden.

  Optimal:
  Das System errechnet das optimale Y-Intervall selbst, so daß die ganze Kurve zu sehen ist.
- Unter Punkt **Zeichnen** wählt man, ob die Funktion selbst, ihre **Ableitung** oder ihr **Integral** gezeichnet werden soll. Das Integral wird willkürlich am linken Rand des X-Intervalls zu Null angenommen, so daß man nur einen rein qualitativen Verlauf erhält.
- Unter Punkt **Fenster** kann man jetzt ein Fenster öffnen, in dem die Funktion dargestellt wird. Es ist dann möglich, das Fenster wie üblich zu vergrößern und zu verkleinern oder auf dem Bildschirm zu verschieben. Sind bereits alle Fenster geöffnet, kann man entweder eins schließen und gleich neu öffnen oder die Anzeige der Funktion durch einen **Doppelklick** im gewünschten Fenster erzwingen. Im Titelbalken des Fensters steht immer, was momentan dargestellt wird, und direkt über dem Graphen der Funktion steht der angezeigte X-, bzw. Y-Bereich.

2. Das Programm

Jetzt aber zum Programm. In Bild 2 sehen Sie die Aufteilung des ganzen in einzelne Module. Das Hauptmodul **FUNPLOT** ist in Listing 1 in voller Länge abgedruckt, da es eigentlich lauter Dinge enthält, die für die GEM Programmierung wichtig sind.

INIGEM ist dasselbe wie in der letzten Folge und im Teil 3 des C Kurses. Es übernimmt, wie üblich, solche Dinge wie Öffnen und Schließen von Workstations etc.

PLOT enthält die Funktionen zum Zeichnen der Kurve. Es soll hier nicht näher betrachtet werden, da es außer ein paar einfachen VDI Aufrufen nichts Aufregendes hergibt.

MENU enthält alle Funktionen zur Interaktion über die Menüleiste. Ich werde später noch detaillierter darauf eingehen.

Das Modul **DIALOG** managt die Interaktion mit den verschiedenen Dialogboxen, da sie zusammen mit den Menüs das Hauptthema dieser Folge darstellen, werde ich später noch intensiv darauf eingehen.

Von **Dialog** aus wird der Parser aufgerufen. Er versteckt sich im Modul **PARSE** und stellt die Mechanismen bereit, die notwendig sind, um solche Monstren wie

sin(x + cos(sqrt(abs(x ★ x + 10)))

zu verarbeiten. Leider paßt der Parser nicht zum Thema des Kurses, so daß ich an dieser Stelle nicht weiter darauf eingehen werde. Sollte genügend Interesse im Leserkreis vorhanden sein, könnte zu diesem Thema gelegentlich ein eigener Artikel in der ST Computer erscheinen.

Wer sich selbst einmal an der Thematik versuchen will, findet im berühmten „Drachenbuch" von Aho, Sethi, Ullmann (Compilers, Principles, Techniques and Tools, Addison Wesley) viele Anregungen.

In Listing 4 habe ich die kontextfreie Grammatik, nach der ich die eingegebenen Funktionen abarbeite, angegeben. Aus ihr können Sie direkt ablesen, welche Ausdrücke das Programm verarbeiten kann. Dieser Teil des Programms läßt sich sehr leicht erweitern; besorgen Sie sich einfach den Quelltext und schauen Sie sich an, wie ich es gemacht habe.

```
/**********************************************************************/
/* ATARI ST Funktionenplotter V1.0      05.09.86                      */
/*                                                                    */
/* geschrieben von Thomas Weinstein                                   */
/*                 Koenigsberger Str. 29d                             */
/*                 7500 Karlsruhe 1                                   */
/*                 Tel. (0721) 685366                                 */
/*                                                                    */
/* MODULE:  FUNPLOT.C                                                 */
/*                                                                    */
/**********************************************************************/

/**********************************************************************/
/* INCLUDE FILES                                                      */
/**********************************************************************/

#include <define.h>
#include <obdefs.h>
#include <gemdefs.h>
#include "funplot.h"
#include "globals.h"

#ifdef MEGAMAX              /* Ist bei mir in define.h vereinbart */
overlay "FUNPLOT"
#endif

/**********************************************************************/
/* DEFINES                                                            */
/**********************************************************************/

#define  MIN_WIDTH    (15 * gl_wchar)
#define  MIN_HEIGHT   (5 * gl_hchar)
#define  WI_KIND      (CLOSER|NAME|SIZER|MOVER)

/**********************************************************************/

extern int   gl_apid;
extern int   handle;
extern int   phys_handle;

/**********************************************************************/
/* GLOBAL VARIABLES                                                   */
/**********************************************************************/

/*********************************************/
/* Globale Variable fuer virtuelle Workstation */
/*********************************************/

int   gl_hchar;
int   gl_wchar;
int   gl_wbox;
int   gl_hbox;                       /* system sizes             */
int   xwork,ywork,wwork,hwork;       /* Groesst mögliche Workarea */

/************************************/
/* Globale Variable fuer Event Handling */
/************************************/

int   msgbuff[8];               /* Ereignispuffer                   */
int   mx,my;                    /* Maus x and y Position.           */
int   butdown;                  /* Zustand der Mausknoepfe          */
int   ret;                      /* Dummy variable                   */
int   quit;                     /* Wenn True wird Fontedit beendet  */

/**************************************/
/* Globale Variable fuer Objektverwaltung */
/**************************************/

OBJECT   *menuaddr;          /* Adresse des Menueleistenbaums           */
OBJECT   *infoaddr;          /* Adresse des Deskinfo Dialogs            */
OBJECT   *funinaddr;         /* Adresse des Funin Dialogs               */
OBJECT   *xnewaddr;          /* Adresse des Xinter Dialogs              */
OBJECT   *scaleaddr;         /* Adresse des Scale Dialogs               */

/***************************************/
/* Globale Variable fuer Fensterverwaltung */
/***************************************/

int      act_window;         /* Aktives Fenster                         */
WI_DESC  windows[4];         /* Datenstruktur zur Fensterverwaltung     */
char     act_function[80]; /* Ueber Dialogbox eingestellt Funktion      */

POINT    points[4][MAXVALS]; /* Punkteliste zu jedem Fenster der Einfachheit */
                             /* halber statisch angelegt.                    */

```

Es bleiben noch die beiden Module **CALC** und **GETVAL. CALC** berechnet die eigentlichen Funktionswerte und bedient sich dabei des Moduls **GETVAL**, das eine sogenannte virtuelle Stackmaschine realisiert, die den durch **PARSE** gewonnenen Postfixausdruck abarbeitet.

2.1 Die Fensterverwaltung

Nun in die Details. In **FUNPLOT** finden Sie die Teile wieder, die wir beim letzten Mal schon ausführlich besprochen haben. Die **main()** Funktion, die alles initialisiert und die **multi()** Schleife, in der das Programm dann bis zum Schluß läuft und auf Ereignisse wartet.

Was neu ist, ist die Verwendung von vier Fenstern statt nur einem, wie im **Browser** Programm. Um mehrere Fenster in den Griff zu bekommen, sollte man sich eine geeignete Datenstruktur überlegen, die die Verwaltung möglichst einfach macht. Schauen Sie sich dazu bitte Listing 2 an. Es enthält außer einigen wichtigen Konstantendefinitionen die Definitionen für zwei wichtige Structs, die Sie im ganzen Programm überall wiederfinden werden. Ganz grundlegend zuerst der struct POINT. Er steht für einen Punkt in der X-Y-Ebene.

Wichtiger ist aber der Typ **WI_DESC**. In dieser Struktur werden alle Informationen zu einem Fenster zusammengefaßt. An erster Stelle steht die Ihnen schon wohlbekannte Window Handle, die das jeweilige Fenster identifiziert. Dann folgen mit **xw, yw, ww, hw** die Abmessungen des Arbeitsbereiches des Fensters. **points** zeigt entweder auf ein Array von Koordinaten, steht also für die dem Fenster aktuell zugeordnete Funktion, oder hat den Wert Null. Dieser Wert steht für die Information, daß dem Fenster momentan keine Funktion zugeordnet ist. Durch das Aufbewahren der Funktionswerte kann man sich die fortwährende Neuberechnung sparen.

Außerdem enthält **WI_DESC** noch Angaben zum eingestellten Skalierungsmodus und, falls der Modus **BESTFIT** eingestellt ist, den minimalen und maximalen Y-Wert. Im Array Title wird die Fenstertitelzeile gehalten.

Da wir vier Fenster verwalten wollen, legen wir uns ein Array aus vier solcher Strukturen an, es trägt den Namen **windows[]**.

Diese Datenstruktur wird am Anfang

```
/******************************************/
/* Globale Variable fuer Plot            */
/******************************************/

int   scalemode = BESTFIT;    /* Art der Skalierung                                    */
int   show_what = FUNCTION;   /* Was wird gezeigt (FUNK, ABL, INTEGRAL)                */

float xfrom = -3.0,         /* Defaultbereiche in X- und Y-Richtung                   */
      yfrom = -3.0,
      xto   =  3.0,
      yto   =  3.0;
                            /* Kleinster resp. groesster Y Wert                      */
float ymin, ymax;

float a,b,c,d;

char  function[41];

int   width = 1;            /* Liniendicke                                            */

/*****************************************************************************/
/* Open Windows                                                              */
/*                                                                           */
/* Oeffnet Fenster 'index' (0 - 3)                                           */
/*****************************************************************************/
open_window(index)
{
   int   hndl;
   int   x,y,w,h;

   x = windows[index].xw;
   y = windows[index].yw;
   w = windows[index].ww;
   h = windows[index].hw;

   if ((hndl = windows[index].wi_hndl =
             wind_create(WI_KIND,xwork,ywork,wwork,hwork)) < 0)
   {
     form_alert(1,"[1][Kein Fenster mehr|verfügbar][   ]");
     return;
   }

   wind_set(hndl, WF_NAME,act_function,0,0);

   hide_mouse();
   wind_open(hndl,x,y,w,h);
   wind_get(hndl,WF_WORKXYWH,&x,&y,&w,&h);

   windows[index].xw      = x;
   windows[index].yw      = y;
   windows[index].ww      = w;
   windows[index].hw      = h;
   windows[index].points = NULL; /* NULL bedeutet: Diesem Fenster ist im   */
                                 /* Moment keine Funktion zugeordnet.      */

   clear_area(x,y,w,h);          /* Arbeitsbereich loeschen                */

   show_mouse();
} /* open_window() */

/*********************************************************/
/* find and redraw all clipping rectangles             */
/*********************************************************/
do_redraw(wi_h,xc,yc,wc,hc)
int wi_h,xc,yc,wc,hc;
{
GRECT t1,t2;

   hide_mouse();
   wind_update(TRUE);
   t2.g_x=xc;
   t2.g_y=yc;
   t2.g_w=wc;
   t2.g_h=hc;
   wind_get(wi_h,WF_FIRSTXYWH,&t1.g_x,&t1.g_y,&t1.g_w,&t1.g_h);
   while (t1.g_w && t1.g_h)
   {
     if (rc_intersect(&t2,&t1))
     {
       set_clip(t1.g_x,t1.g_y,t1.g_w,t1.g_h);
       redraw(wi_h,t1.g_x,t1.g_y,t1.g_w,t1.g_h);
     }
     wind_get(wi_h,WF_NEXTXYWH,&t1.g_x,&t1.g_y,&t1.g_w,&t1.g_h);
   }
   wind_update(FALSE);
   show_mouse();
}  /* do_redraw */
```

```
173 :
174 : /*
175 :  * Initialisieren der Fensterdatenstruktur
176 :  */
177 : init_wi_desc()
178 : {
179 :    int i;
180 :
181 :    for (i = 0; i < 4; i++)
182 :             init_window(i);
183 :
184 : }
185 :
186 : /*
187 :  * Initialisiert das Fensterobjekt i
188 :  */
189 : init_window(i)
190 : int i;
191 : {
192 :    windows[i].wi_hndl    = NO_WINDOW;
193 :    windows[i].points     = NULL;
194 :    windows[i].xw         = xwork + (i % 2) * wwork/2;
195 :    windows[i].yw         = ywork + (i > 1) * hwork/2;
196 :    windows[i].ww         = wwork/2;
197 :    windows[i].hw         = hwork/2;
198 :    windows[i].scalemode = BESTFIT;
199 : }
200 :
201 : /*
202 :  * Holt den Index der zu hndl gehoerigen Datenstruktur.
203 :  * Falls nicht existent wird -1 geliefert.
204 :  */
205 : int get_index(hndl)
206 : int    hndl;
207 : {
208 :    int i;
209 :
210 :    for (i = 0; i < 4; i++)
211 :       if (windows[i].wi_hndl == hndl) break;
212 :
213 :    return((i == 4) ? -1 : i);
214 : }
215 :
216 : /*
217 :  * Beendet Programm
218 :  */
219 : ex_app()
220 : {
221 :    close_all_windows();
222 :    v_clsvwk(handle);
223 :    appl_exit();
224 : }  /* ex_app() */
225 :
226 : /*
227 :  * Schliesst alle offenen Fenster
228 :  */
229 : close_all_windows()
230 : {
231 :    int i;
232 :
233 :    for (i = 0; i < 4; i++)
234 :       if (windows[i].wi_hndl != NO_WINDOW)
235 :       {
236 :          wind_close(windows[i].wi_hndl);
237 :          wind_delete(windows[i].wi_hndl);
238 :          windows[i].wi_hndl = NO_WINDOW;
239 :          windows[i].points  = NULL;
240 :       }
241 : }
242 :
243 : /***************************************************************************/
244 : /*       FUNPLOT INITIALISIERUNG BIS ZUM ERSTEN MULTI AUFRUF              */
245 : /***************************************************************************/
246 : main()
247 : {
248 :
249 :    appl_init();
250 :    phys_handle=graf_handle(&gl_wchar,&gl_hchar,&gl_wbox,&gl_hbox);
251 :    wind_get(0, WF_WORKXYWH, &xwork, &ywork, &wwork, &hwork);
252 :    open_vwork();
253 :
254 :    if (!rsrc_load("funplot.rsc"))
255 :    {
256 :       form_alert(1,"[1][Kann RSC File nicht finden][ABBRUCH]");
257 :       close_vwork();
258 :       appl_exit();
259 :    }
```

auf Defaultwerte initialisiert – Listing 1, Zeile 177, Funktion **init_wi_desc()** – und dann an allen Stellen, an denen das Fenster verändert wird, auf den neuesten Stand gebracht. Die Information wird dann von der Funktion **plot_function()** benutzt, um die Kurve an die richtige Stelle des Bildschirms zu zeichnen.

Die Verwaltung der Fenster, was Vergrößern, Verkleinern, Verschieben oder Redraw betrifft, ist die gleiche wie im Browser Beispiel. Der einzige Unterschied ist der, daß man jetzt beim Auftreten eines Ereignisses (s. **multi()**) zuerst feststellen muß, welches der vier möglichen Fenster der Auslöser ist. Alles andere, was ich Ihnen im Teil 1 über die Rechtecklisten und das Neuzeichnen der Fenster gesagt habe, gilt hier natürlich ebenso.

2.2 Die Resourcenverwaltung

Das Konzept der Resourcen stellt ein wichtiges Hilfsmittel in GEM dar. Mein Englischwörterbuch übersetzt Resource mit Mittel oder Hilfsmittel. Und das sind sie auch, nämlich Hilfsmittel, die GEM dem Anwendungsprogrammierer zur Verfügung stellt um ihm die Interakton mit dem späteren Benutzer des Programms zu ermöglichen. Die bekanntesten Resourcen sind die Menüleiste, die Dialogboxen und die Alarmboxen.

Wie stellen sich die Resourcen für den Programmierer dar? In der Dokumentation zu GEM werden sie als Bäume (trees) bezeichnet. In Wirklichkeit sind es jedoch gerichtete Graphen, was aber im Moment nur von untergeordneter Bedeutung ist, daher werde ich sie weiterhin als Bäume bezeichnen. Falls Sie sich nichts unter Bäumen vorstellen können (zumindest im Zusammenhang mit Computern) sollten Sie sich an den Teil 4 meines C Kurses erinnern, in dem ich für das Crossreferenzprogramm ebenfalls einen Baum aufgebaut habe.

Bei GEM besteht ein solcher Baum aus Objekten, die untereinander verzeigert sind. Als Beispiel sehen wir uns die Dialogbox zur Funktionseingabe an (Bild 3). Die Wurzel, also das oberste Objekt des Baumes ist die Dialogbox selbst, das heißt der Rahmen, der um das ganze Geschehen gezogen wird. Von der Dialogbox aus gehen jetzt Verweise auf weitere Objekte, z. B. auf Strings („FUNKTIONSEINGABE“) oder Buttons („Alles in Ordnung“, „Nichts ändern“) oder

editierbare Textfelder wie etwa das Feld, in dem der Benutzer die neue Funktion eingibt. Der Gag an der Sache ist jetzt der: Ein Objekt ist programmintern eine C Struktur (Sie wissen ja, daß GEM in C geschrieben wurde) und kann daher mit den üblichen Methoden für Strukturen bearbeitet werden.
Jedes Objekt enthält Angaben über seine Art (Box, String, Button...) und über die relative Lage zum übergeordneten Objekt. Diese Lage wird einmal durch Koordinaten ausgedrückt, die relativ zur linken oberen Ecke des übergeordneten Objekts angegeben sind, andererseits durch die Verzeigerung der Objekte untereinander. Durch diese Relativbeziehung zwischen den Objekten ist es sehr leicht möglich, Objekte zu verschieben. Man muß nur die Koordinaten des obersten Objektes ändern, die übrigens auch relativ sind, nämlich zur linken oberen Ecke des Bildschirms, und automatisch verschieben sich alle anderen untergeordneten Objekte mit. Je nach Art des Objekts gehen von der Objektdatenstruktur Zeiger auf weitere Strukturen aus.

Bei einem editierbaren Textfeld gibt es z. B. einen Zeiger auf eine sogenannte TEDINFO Struktur in der Angaben über den Text, seine Länge, erlaubte Eingabezeichen etc. stehen. Ist das Objekt ein ICON, gibt es einen Zeiger auf eine ICONBLK Struktur, in der das Bitmuster des Icons und seiner Maske steht.

In der nächsten Folge des GEM Kurses will ich ganz detailliert auf die Beeinflussungsmöglichkeiten eingehen, die man durch Zugriff auf diese internen Strukturen hat.

In dieser Folge werden wir aber die Dienste des Resource Construction Sets (RCS) benutzen, der es uns erlaubt, die Resourcen interaktiv zu erzeugen und der die ganze Koordinatenrechnerei übernimmt.

Der RCS erzeugt eine Resourcedatei, in der die oben geschilderten Bäume und Objekte in einer geräte-unabhängigen Darstellung stehen. Die Aufgabe des Programmierers ist es nun, mit Hilfe der Funktion **rsrc_load()** diese Datei zu laden. Sie finden diese Stelle in Listing 1 ab Zeile 254. **rsc_load()** reserviert Platz für die Resourcedaten, lädt die Datei und nimmt alle Umrechnungen von Einheiten und Adressen vor, die notwendig sind. Der **rsrc_load()** Aufruf liefert 0 zurück, falls ein Fehler aufgetreten ist.

```
260 :
261 :     rsrc_gaddr(R_TREE,MENU,&menuaddr);    /* Initialisieren der        */
262 :     rsrc_gaddr(R_TREE,INFO,&infoaddr);    /* Objektadressen            */
263 :     rsrc_gaddr(R_TREE,FUNIN,&funinaddr);
264 :     rsrc_gaddr(R_TREE,XINTER,&xnewaddr);
265 :     rsrc_gaddr(R_TREE,SCALING,&scaleaddr);
266 :
267 :     /* Dieser kleine Trick sorgt dafuer, dass der Cursor bei der  */
268 :     /* ersten Funktionseingabe ganz links steht.                  */
269 :     get_string(funinaddr,FUNDEF,function);
270 :     sprintf(function,"@040                                  ");
271 :     set_string(funinaddr,FUNDEF,function);
272 :
273 :     act_function[0] = '\0';
274 :     function[0]     = '\0';
275 :
276 :     init_wi_desc();
277 :
278 :     menu_bar(menuaddr,1);                    /* Anzeigen der Menueleiste  */
279 :
280 :     graf_mouse(ARROW,0x0L);                  /* Maus auf Pfeil umschalten */
281 :
282 :     butdown=TRUE;
283 :
284 :     multi();
285 : }
286 :
287 :
288 : /**************************************************************************
289 : /* Koordiniert alle Benutzeraktionen                                      */
290 : /**************************************************************************
291 : multi()
292 : {
293 :     int event, no_clicks, olo;
294 :     int i, x,y,w,h;
295 :     int hndl;
296 :
297 :     do
298 :     {
299 :         event = evnt_multi(MU_MESAG|MU_BUTTON,
300 :                            3,1,butdown,
301 :                            0,0,0,0,0,
302 :                            0,0,0,0,0,
303 :                            msgbuff,0,0,&mx,&my,&ret,&ret,&ret,&no_clicks);
304 :
305 :         wind_update(TRUE);
306 :
307 :         if (event & MU_MESAG)
308 :            switch (msgbuff[0])
309 :            {
310 :
311 :               case MN_SELECTED:
312 :                  hndl_menu(msgbuff[3],msgbuff[4]);
313 :                  menu_tnormal(menuaddr,msgbuff[3],1);
314 :               break;
315 :
316 :               case WM_REDRAW:
317 :                  do_redraw(msgbuff[3],msgbuff[4],
318 :                            msgbuff[5],msgbuff[6],msgbuff[7]);
319 :               break;
320 :
321 :               case WM_NEWTOP:
322 :                  if ((i = get_index(msgbuff[3])) != -1)
323 :                  {
324 :                      wind_set(msgbuff[3],WF_TOP,0,0,0,0);
325 :                      act_window = i;
326 :                      windows[i].points = NULL;
327 :                      plot_function(i);
328 :                  }
329 :               break;
330 :
331 :               case WM_TOPPED:
332 :                  if ((i = get_index(msgbuff[3])) != -1)
333 :                  {
334 :                      wind_set(msgbuff[3],WF_TOP,0,0,0,0);
335 :                      act_window = i;
336 :                  }
337 :               break;
338 :
339 :               case WM_SIZED:
340 :               case WM_MOVED:
341 :                  if ((i = get_index(msgbuff[3])) != -1)
342 :                  {
343 :                      hndl = windows[i].wi_hndl;
344 :
345 :                      /* Ist Veraenderung zulaessig ? */
346 :                      if(msgbuff[6]<MIN_WIDTH)  msgbuff[6] = MIN_WIDTH;
```

```
                        if(msgbuff[7]<MIN_HEIGHT) msgbuff[7] = MIN_HEIGHT;

                        /* Setze neue Fenstergroesse/position */
                        wind_set(hndl,WF_CURRXYWH,msgbuff[4],msgbuff[5],
                                                  msgbuff[6],msgbuff[7]);

                        /* Groesse des neuen Arbeitsbereichs feststellen */
                        wind_get(hndl,WF_WORKXYWH,&x,&y,&w,&h);

                        windows[i].xw = x;
                        windows[i].yw = y;
                        windows[i].ww = w;
                        windows[i].hw = h;
                    }
                 break;

                 case WM_CLOSED:
                    if ((i = get_index(msgbuff[3])) != -1)
                    {
                        wind_close(msgbuff[3]);
                        wind_delete(msgbuff[3]);
                        windows[i].wi_hndl = NO_WINDOW;
                    }
                 break;

           } /* switch (msgbuff[0]) */

           wind_update(FALSE);

           if (event & MU_BUTTON)
              if (no_clicks == 2)
              {
                 if (hndl = wind_find(mx,my))
                 {
                    i = get_index(hndl);
                    windows[i].points = NULL;
                    clear_area(windows[i].xw,
                               windows[i].yw,
                               windows[i].ww,
                               windows[i].hw);
                    plot_function(i);
                 }
              }
              else
              if (no_clicks > 2) /* Kleiner Scherz am Rande */
                    message("Nur nicht so stürmisch !");

    } while( !quit );

    ex_app();
 }  /* multi */

/*
 * Zeichnet sichtbaren Arbeitsbereich von Fenster mit hndl neu.
 */
redraw(hndl,x,y,w,h)
int x,y,w,h;
{
    int i;

    if ((i = get_index(hndl)) != -1)
    {
       clear_area(x,y,w,h);
       plot_function(i);
    }
}

/*
 * Loescht rechteckigen Bildschirmbereich
 */
clear_area(x,y,w,h)
int x,y,w,h;
{
    int   pxy[4];

    pxy[0] = x;
    pxy[1] = y;
    pxy[2] = x + w - 1;
    pxy[3] = y + h - 1;

    hide_mouse();
    vsf_color(handle,0);
    set_clip(x,y,w,h);
    v_bar(handle,pxy);
    show_mouse();

} /* clear_area */
```

Wenn **rsrc__load()** erfolgreich zurückkehrt, ist die Resourcedatei fest im Programm verankert und kann benutzt werden. Der nächste Schrit ist, sich die Adressen der Objektbäume zu besorgen. Dies übernimmt die Funktion **rsrc__gaddr()**.

Es ist übrigens ein grober Fehler und überdies schlechter Programmierstil, den Datentyp der Objektadressen als **long** anzugeben. Der korrekte Typ ist **OBJECT ★** und normalerweise in **obdefs.h** vereinbart. Durch die Vereinbarung eines long-Werts gibt man viele Beeinflussungsmöglichkeiten auf.

Als Beispiel schauen wir uns das Holen der Adresse zum Funktionseingabedialog an. Zuerst wurde in Zeile 71 von **FUNPLOT** folgende Vereinbarung getroffen:

OBJECT ★funinaddr;

Damit hat man eine Variable vereinbart, die einen Zeiger auf ein Object vom Typ **OBJECT** speichern kann. Hinter **OBJECT** versteckt sich natürlich genau die Datenstruktur, die ich weiter oben schon einige Male angesprochen habe.

Jetzt können wir dieser Variablen in Zeile 263 die Adresse des Dialogbaumes zuweisen. Dies ist natürlich genau die Adresse des obersten Objektes in der Dialogbox, nämlich des umrahmenden Rechtecks:

rsrc__gaddr(R__TREE, FUNIN,&funinaddr);

R__TREE ist eine Konstante, die wiederum in **obdefs.h** vereinbart ist und der Funktion sagt, daß sie die Adresse eines Baumes liefern soll. Die nächste Konstante **FUNIN** wurde vom RCS in der Datei **funplot.h** angelegt und ist der Index in das Array aller Baumadressen.

Klar dürfte sein, daß wir die Adresse der Variablen **funinaddr** übergeben müssen. Sie erinnern sich sicher an die Grundregel: Wenn eine Variable in einer Funktion geändert werden soll, muß man ihre Adresse als Parameter übergeben.

Nachdem wir die Resource geladen und uns die Adressen der einzelnen Bäume geholt haben, müssen wir jetzt nur noch wissen, wie man den Dialog selbst programmiert.

Wir werden uns wieder die Funktionseingabe als Beispiel ansehen. Schauen Sie sich dazu zuerst einmal in Listing 3 die Zeilen 49–89 an. Die Funktion, die den

Dialog managt, heißt **funin__dialog()**.

Zuerst wird eine Funktion **hndl__dialog()** aufgerufen, in der ich alles, was zum eigentlichen Dialog gehört, verborgen habe – Näheres gleich. Diese Funktion liefert als Ergebnis zurück, welcher der Exit-Buttons betätigt wurde (OK oder CANCEL). Die Exit-Buttons stellen die einzige Möglichkeit dar, einen Dialog wieder zu verlassen. Solange kein Exit-Button angeklickt wurde, kann der Benutzer beliebig in der Dialogbox herumändern und editieren.

Einer der Exit-Buttons kann als Default deklariert werden (natürlich alles im RCS) und wird dann beim Drücken der Return-Taste ausgelöst. Einen Default Exit-Button erkennt man übrigens daran, daß er einen breiteren Rand hat.

Im **funin__dialog()** müssen nun die beiden möglichen Exit__Buttons abgehandelt werden. Wenn der Benutzer den „Alles in Ordnung" Button angeklickt hat, müssen die geänderten Werte aus der Objektdatenstruktur herausgeholt und dem Programm bekannt gemacht werden. Hat er andrerseits den „Nichts ändern" Knopf betätigt, müssen die eventuell geänderten Werte wieder in den alten Zustand versetzt werden. Aus Platzmangel werde ich nur den Fall, daß eine geänderte Funktionseingabezeile geholt werden muß, näher behandeln, alle anderen Fälle laufen entsprechend ab.

In **funin__dialog()** wird in Zeile 69 folgender Aufruf gemacht:

get__string(funinaddr, FUNDEF,function);

Funinaddr ist, wie Sie ja schon wissen, die Adresse des Objektbaumes, der den Funktionseingabedialog darstellt. **FUNDEF** ist der Index des Objektes, das für das editierbare Textfeld mit der Funktionszeile steht. **Function** ist ein Rückgabeparameter, in dem Funktionsstring übergeben wird.

In den Zeilen 93–108 können Sie sehen, wie **get__string()** realisiert wurde. Da die Eingabezeile ein editierbares Textfeld ist, zeigt das Feld **ob__spec** in der Objektdatenstruktur der Eingabezeile auf eine schon oben erwähnte **TEDINFO** Struktur, die ihrerseits im Feld **te__ptext** die neue Funktionszeile als String enthält. Man muß sich also zweistufig an die gewünschte Information herantasten.

text__desc = (TEDINFO ★) funinaddr[item].ob__spec;

```
434 :
435 : /*
436 :  * Eigentliche Zeichenfunktion. Veranlasst alles um Kurve im Fenster
437 :  * darzustellen.
438 :  */
439 : plot_function(index)
440 : int index;
441 : {
442 :     int    i,x,y,w,h,s;
443 :     char  *what = "";
444 :     POINT *p;
445 :
446 :     if (*function == '\0') return;
447 :
448 :
449 :     x     = windows[index].xw;
450 :     y     = windows[index].yw;
451 :     w     = windows[index].ww;
452 :     h     = windows[index].hw;
453 :     s     = windows[index].scalemode;
454 :
455 :     if (windows[index].points == NULL) /* Fenster hat keine Funktion zugeord. */
456 :     {
457 :             windows[index].points = points[index];
458 :
459 :             if (!calc_vals(index)) return; /* Berechnen der Werte */
460 :
461 :             s = windows[index].scalemode = scalemode;
462 :
463 :             /* Was soll gezeichnet werden ? */
464 :             if (show_what == DERIVATE)
465 :             {
466 :                     what = "D/DX ";
467 :                     calc_der(index);     /* Berechne Ableitung */
468 :             }
469 :             else
470 :             if (show_what == INTEGRAL)
471 :             {
472 :                     what = "INT von";
473 :                     calc_int(index);    /*
474 :                                          * Berechne Integral (rein qualitativ).
475 :                                          * Linke Intervallgrenze wird zu 0 an-
476 :                                          * genommen.
477 :                                          */
478 :             }
479 :             else
480 :                     what = "f(x) = ";
481 :
482 :             p = windows[index].points;
483 :
484 :             ymin = ymax = p[0].y;
485 :
486 :             /* ymin, ymax nur bei Bestfit Methode berechnen */
487 :             if (s == BESTFIT)
488 :                 for (i = 0; i < MAXVALS; i++)
489 :                 {
490 :
491 :                     if ((p[i].y - ymin) < 0.0) ymin = p[i].y;
492 :                     else
493 :                     if ((ymax - p[i].y) < 0.0) ymax = p[i].y;
494 :                 }
495 :             windows[index].ymin = ymin;
496 :             windows[index].ymax = ymax;
497 :
498 :             sprintf(windows[index].title,"%s %s ",what,function);
499 :             wind_set(windows[index].wi_hndl, WF_NAME,windows[index].title,0,0);
500 :     }
501 :     else
502 :     {
503 :             ymin = windows[index].ymin;
504 :             ymax = windows[index].ymax;
505 :     }
506 :
507 :
508 :     hide_mouse();
509 :     plot(MAXVALS,windows[index].points,x,y,w,h,s);  /* s. PLOT.C */
510 :     show_mouse();
511 :
512 : }
513 :
514 : /*
515 :  * Berechnet Ableitung (nicht sonderlich genau).
516 :  * Methode:
517 :  *          f'(x) ~ ( f(x+dx) - f(x) ) / dx
518 :  *
519 :  */
520 : calc_der(index)
```

```
521 : int index;
522 : {
523 :     int i;
524 :     POINT *p = windows[index].points;
525 :
526 :     for (i = 0; i < MAXVALS-1; i++)
527 :        p[i].y = (p[i+1].y - p[i].y) / (p[i+1].x - p[i].x);
528 :
529 :     p[MAXVALS-1].y = p[MAXVALS-2].y;
530 :     p[MAXVALS-1].x = p[MAXVALS-2].x;
531 : }
532 :
533 : /*
534 :  * Berechnet Integral nach Rechteckverfahren. Wert an linker Intervallgrenze
535 :  * wird willkuerlich zu 0 angenommen. Damit rein qualitativer Verlauf.
536 :  */
537 : calc_int(index)
538 : int index;
539 : {
540 :     int i;
541 :     POINT *p = windows[index].points;
542 :
543 :     p[0].y = 0;   /* Anfangswert */
544 :
545 :     for (i = 1; i < MAXVALS; i++)
546 :         p[i].y = p[i-1].y + (p[i].y * (p[i+1].x - p[i].x));
547 :
548 :     p[MAXVALS-1].y = p[MAXVALS-2].y;
549 :     p[MAXVALS-1].x = p[MAXVALS-2].x;
550 : }
551 :
552 : /*
553 :  * Gibt mit Hilfe von form_alert Nachricht an Benutzer
554 :  */
555 : message(m)
556 : char *m;
557 : {
558 :     char show[MAXVALS];
559 :
560 :     sprintf(show,"[1][%s][ OK ]",m);
561 :     form_alert(1,show);
562 : }
563 :
564 : /* ENDE von FUNPLOT.C */
565 :
```

Listing 1

FUNKTIONSEINGABE

f(x,a,b,c,d) =

x*sin(4*x)/(x-4)_______________________

a: 1.0000_ | b: 1.0000_ | c: 1.0000_ | d: 1.0000_

Alles in Ordnung | Nichts ändern

Bild 3: Dialogbox zur Funktionseingabe

text__desc zeigt nun auf die TEDINFO Struktur der Eingabezeile.

text = text__desc→te__ptext;

Damit zeigt **text** auf den gesuchten String, der mit **strcpy()** in den Rückgabeparameter befördert wird.

Beim Zurückschreiben eines alten Wertes passiert im Prinzip das Gleiche, bloß rückwärts. Sie können sich selbst anschauen wie in den Zeilen 110–123.

Nun noch ein kurzer Blick auf die Funktion **hndl__dialog()**. In ihr ist das ganze notwendige Drumherum des eigentlichen Dialogs versteckt. Sie können die Funktion für eigene Anwendungen ungeändert übernehmen. Nur wenn Sie ganz spezielle Dinge tun wollen, kann es notwendig sein, etwas zu ändern.

Die Dialogboxen sind normalerweise ganz in die linke obere Ecke des Bildschirms verschoben. Da man den Dialog aber eigentlich immer in der Mitte haben will, muß man die Funktion **form__center()** bemühen, die einem die Koordinaten entsprechend umrechnet. Achten Sie wieder darauf, die Adressen der Variablen zu übergeben.

Als Nächstes folgt die Funktion **form__dial()**. Sie wird während des ganzen Prozesses viermal aufgerufen und tut ganz verschiedene Dinge, je nachdem, welchen Wert der erste Parameter hat. Beim Aufruf mit **FMD__START** wird der zum Darstellen des Dialogs nötige Platz „gemerkt". Allerdings wird der Platz nicht gerettet, wie man aus manchen Büchern entnehmen kann.

Beim zweiten Aufruf mit **FMD__GROW** wird ein immer größer werdendes Rechteck gezeichnet, also ein rein optisch-ästhetischer Effekt.

Jetzt ist alles bereit, den Dialog tatsächlich zu zeichnen. Das übernimmt die Funktion **objc__draw()**. Es folgt **form__do()**. Hier erfolgt jetzt der eigentliche Dialog. Die Funktion erhält als ersten Parameter die Adresse des Dialogbaums und als zweiten den Index des ersten editierbaren Textfeldes. Es ist wichtig, daß es wirklich das erste ist, sonst gibt's CHAOS. Gibt es keine editierbaren Textfelder im Dialog, wird hier eine Null übergeben.

Form__do() liefert die Exit Condition zurück, und die Aufräumarbeiten macht jetzt wieder **form__dial()**. Zuerst wird mit **FMD__SHRINK** ein schrumpfen-

des Rechteck gemalt und dann mit **FMD__FINISH** die Restaurierung des Bildschirms eingeleitet. Und hier liegt der Hase im Pfeffer: Entgegen verschiedener Spekulationen wird von **form__dial()** der Bildschirm **nicht** neu gezeichnet. Alles, was **form__do()** tut, ist, eine **Redraw**-Nachricht an den Desktopmanager und die Applikation zu schicken. Wenn Ihr Programm nun **Redraw**-Nachrichten ignoriert, weil Sie vielleicht gar keine Fenster offen haben, hat der Bildschirm nachher an Stelle der Dialogbox ein Loch. In solchen Fällen müssen Sie sich also explizit um die Wiederherstellung des Bildschirms kümmern.

Ein wirksamer Trick ist es, unter der Dialogbox ein Fenster aufzumachen oder mit einem Blocktransfer das gefährdete Gebiet zu retten und nachher wieder zurückzuschreiben. Sie können in manchen Fällen auch einfach explizit das zerstörte Gebiet neu zeichnen, falls es nicht zu lange dauert und Sie in der jeweiligen Situation überhaupt noch wissen, was dort war. Falls Sie die letztere Methode wählen, sollten Sie das neu zu zeichnende Gebiet etwa 3 Punkt größer wählen als die Dialogbox war, da sonst ein Rand stehen bleibt.

2.3 Die Menüleiste

Bei der Menüleiste kann ich mich sehr kurz fassen. Zu diesem Thema hat Dirk Owerfeldt in der Oktoberausgabe der ST Computer schon alles gesagt, was es zu sagen gibt. Kritik kann man nur daran üben, daß er die Adressen seiner Objektbäume als long vereinbart (s. o.), andererseits stört es in seinem Programm nicht sonderlich, da die Adressen – außer zum Zeichnen der Boxen – sowieso nicht weiter verwendet werden.

In **FUNPLOT** können Sie sehen, daß man bei größeren Programmen aus Gründen der Übersichtlichkeit am besten das ganze Menühandling in ein separates Modul packt und in der **multi()** Schleife nach Erhalt der **MN__SELECTED** Nachricht nur noch eine entsprechende Funktion aufruft. Ansonsten werden fast alle Möglichkeiten, die die Menüleiste bietet, verwendet, so daß Sie sich die Einzelheiten im Modul **MENU** noch einmal ansehen können. Leider ist hier nicht genug Platz, um das Modul abzudrucken, so daß ich Sie nochmals

```
/*
 * GLOBALS.H
 *
 */

#define  YEQUALX     MEQUAL
#define  YINTERVALL  MINTER
#define  BESTFIT     MFIT

#define  FUNCTION    MFUNC
#define  DERIVATE    MDERIV
#define  INTEGRAL    MINT

#define  NO_WINDOW   -1
#define  MAXVALS     100

/*************************************************************************/
/* TYPEN                                                                 */
/*************************************************************************/

typedef struct point {
    float x,y;
} POINT, PARRAY[];

typedef struct window_desciptor {
            int       wi_hndl;            /* Window Handle               */
            int       xw,yw,ww,hw;        /* Groesse des Arbeitsbereichs*/
            POINT     *points;            /* Wertetabelle der Funktion   */
            int       scalemode;          /* Art der Skalierung          */
            float     ymin,ymax;          /* kleinster, groesster YWert  */
            char      title[40];          /* Titel des Fensters          */
} WI_DESC;


/* Ende von GLOBALS.H */
g
```

Listing 2

```
                                  .
                                  .
                       Ausschnitt aus DIALOG.C
                                  .
                                  .
/*
 * Managt den gesamten Dialog mit einer Dialogbox, einschliesslich Zeichnen
 * des Dialogs und wiederherstellen des Bildschirms am Schluss.
 *
 * EINGABE:
 * addr ist die Adresse eines Dialogbaums
 * edit_item ist der Index des ersten editierbaren Textfeldes im Baum und
 * 0 falls keines vorhanden.
 *
 * AUSGABE:
 * exit_cond des Dialogs, d.h Index des Exit Buttons
 */
int     hndl_dialog(addr,edit_item)
OBJECT   *addr;
int       edit_item;
{
    int    ex_cond;
    int    x,y,w,h;

    x = addr->ob_x;
    y = addr->ob_y;
    w = addr->ob_width;
    h = addr->ob_height;

    form_center(addr,&x,&y,&w,&h);

    form_dial(FMD_START,318,198,4,4,x,y,w,h);
    form_dial(FMD_GROW,318,198,4,4,x,y,w,h);

    objc_draw(addr,0,MAX_DEPTH,x,y,w,h);

    ex_cond = form_do(addr,edit_item);

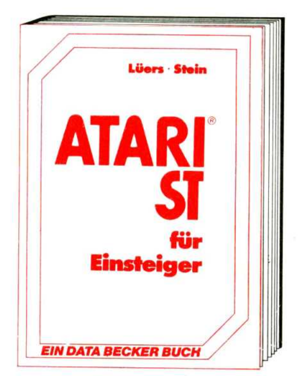
    form_dial(FMD_SHRINK,318,198,4,4,x,y,w,h);
    form_dial(FMD_FINISH,318,198,4,4,x,y,w,h);

    return(ex_cond);

} /* hndl_dialog */

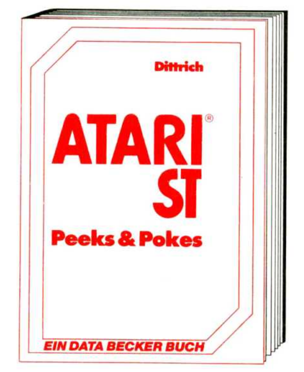

```

BESTSELLER



Das erste Buch zum ATARI ST ist für viele das wichtigste. Denn der richtige Einstieg garantiert später die volle Nutzung dieses Superrechners. ATARI ST für Einsteiger ist eine leichtverständliche Einführung in Handhabung, Einsatz und Programmierung Ihres ST: Von der Tastatur und der Maus hin zum Desktop bis zum ersten BASIC-Befehl und schließlich zu ganzen Programmen. Die Autoren zeigen Ihnen den Weg zum Erfolg mit Ihrem neuen Rechner.

ATARI ST für Einsteiger
262 Seiten, DM 29,–



Peeks & Pokes – zwei Worte, die oft wie ein „Sesam-öffne-dich" wirken. Einbinden von Maschinensprache in BASIC, Joystickabfrage: Dieses Buch zeigt Ihnen das Wie. Bei der Gelegenheit lernen Sie auch gleich eine ganze Menge über die Internas Ihres ST: Betriebssystem, Pointer, Interpreter, Stacks… Dieses Buch ist also nicht nur eine praktische Hilfe in vielen Situationen, sondern auch der erste Schritt in Richtung Maschinensprache.

PEEKS & POKES zum ATARI ST
201 Seiten, DM 29,–



Sie können BASIC? Sie wollen C lernen? Hier ist der schnellste Weg dazu: Von BASIC zu C auf dem ATARI ST. Ausgehend von einfachen BASIC-Programmen wird der entsprechende C-Code entwickelt. Bis hin zu den weiterführenden Sprachelementen wird dabei alles so erklärt, daß Umsteigen ganz leicht fällt. Sogar ein Extra-Anhang mit den typischen Fehlern von BASIC-Umsteigern findet sich. Aber wir sind sicher, daß Sie diese Hilfe nicht allzulange in Anspruch nehmen müssen.

Von BASIC zu C
336 Seiten, DM 39,–

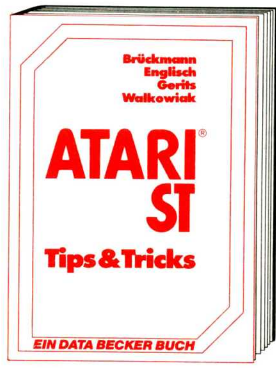
Das Informationspaket zum ATARI ST. Geschnürt vom bewährten INTERN-Team Gerits, Englisch, Brückmann. Ein paar Stichworte aus dem Inhalt: Der 68000-Prozessor, der I/O-Controller MFP 68901, der Soundchip YM-2149, alles über die Schnittstellen des ST und XBIOS, kommentiertes BIOS-Listing, wichtige Systemadressen, MIDI-Interface, Grafikaufbau – dieser kleine Ausschnitt reicht sicher, um klar zu machen: Das ist Pflichtlektüre für ST-Profis.

ATARI ST Intern
Hardcover, 506 Seiten, DM 69,–



Endlich eine umfassende Analyse zu Floppy und Harddisk des ATARI ST. Ein Buch, das sogar die Profis noch erstaunen wird – aber trotzdem auch für den Einsteiger geeignet ist. Dafür sorgen viele ausführliche Beispiele. Controllerbeschreibung, Programmierung unter TOS, BIOS und XBIOS, Boot-Sektor, Programmformate auf Diskette und Festplatte, RAM-Disk – in diesem Buch findet sich alles, was dazugehört.

ATARI ST Floppy und Harddisk
Hardcover, 522 Seiten, DM 59,–



Randvoll mit guten Ideen: Das ist unser Tips und Tricks zum ATARI ST. Sie suchen einen Druckerspooler? In diesem Buch finden Sie ihn. Sie brauchen eine Farb-Hardcopy-Routine? Bitteschön. Eine RAM-Disk anlegen? Nichts leichter als das. Viele neue Routinen, die problemlos in eigene Programme eingebunden werden können, warten nur darauf, daß Sie sie ausprobieren.

ATARI ST Tips & Tricks
Hardcover,
352 Seiten
DM 49,–

NEU: ST-INFO

Mehr über unser großes Angebot aktueller Bücher und Programme zum ATARI ST enthält unser neues ST-INFO, das wir Ihnen gerne zusenden.

DATA BECKER

Merowingerstr. 30 · 4000 Düsseldorf · Tel. (0211) 310010

BESTELL-COUPON

Einsenden an: DATA BECKER · Merowingerstr. 30 · 4000 Düsseldorf 1

Bitte senden Sie mir:

zzgl. DM 5,– Versandkosten

☐ per Nachnahme ☐ Verrechnungsscheck liegt bei

Name

Straße

Ort

ST 17

auf den Verlag verweisen muß, der den gesamten (ziemlich ausführlich kommentierten) Quelltext auf der Monatsdiskette vorliegen hat.

In der nächsten Folge will ich Ihnen zeigen, wie man sich eigene Objekte kreieren und verwalten kann.

Und nun viel Spaß (engl. ~ FUN) beim plotten von Funktionen.

```
                                     .
                                     .
/*
 * Dialog zum Einlesen einer neuen Funktion
 */
funin_dialog()
{
   int   ex_cond;

   ex_cond = hndl_dialog(funinaddr,FUNDEF);

   if (ex_cond == FDEFOK)   /* OK Button wurde betaetigt. Neue Werte holen */
   {
      funinaddr[FDEFOK].ob_state &= ~SELECTED;
      get_float(funinaddr,APARAM,&a);
      identifiers[0] = a;
      get_float(funinaddr,BPARAM,&b);
      identifiers[1] = b;
      get_float(funinaddr,CPARAM,&c);
      identifiers[2] = c;
      get_float(funinaddr,DPARAM,&d);
      identifiers[3] = d;
      get_string(funinaddr,FUNDEF,function);

      /* Hier folgt der Aufruf an den PARSER */
      if (parse(function))
      {
         form_alert(1,"[1][Die Funktion ist|syntaktisch nicht korrekt][ OK ]");
         *function = '\0';
      }

   }
   else   /* CANCEL Button wurde betaetigt. Alte Werte beibehalten */
   {
      funinaddr[FDEFCAN].ob_state &= ~SELECTED;
      set_float(funinaddr,APARAM,a);
      set_float(funinaddr,BPARAM,b);
      set_float(funinaddr,CPARAM,c);
      set_float(funinaddr,DPARAM,d);
      set_string(funinaddr,FUNDEF,function);
      windows[act_window].points = NULL;
   }
}
                                     .
                                     .
                                     .
/*
 *  Liefert in string die Adresse des Textstrings im Dialog addr mit Index
 *  item. ACHTUNG ! Die Funktion behandelt nur editierbare Textfelder
 */
get_string(addr,item,string)
OBJECT    *addr;
int        item;
char      *string;
{
   TEDINFO  *text_desc;
   char     *text;

   text_desc = (TEDINFO *) addr[item].ob_spec;
   text = text_desc->te_ptext;
   strcpy(string,text);
}

/*
 * Gegenstueck zu get_string. Setzt die Adresse des Textes, des editierbaren
 * Textfeldes item im Dialog addr auf string.
 */
set_string(addr,item,string)
OBJECT    *addr;
int        item;
char      *string;
{
   TEDINFO  *text_desc;

   text_desc = (TEDINFO *) addr[item].ob_spec;
   strcpy(text_desc->te_ptext,string);
}
                                     .
                                     .
                                     .
/
```

Listing 3

XLISP

Künstliche Intelligenz zum Nulltarif

Probleme aus dem Bereich der künstlichen Intelligenz wurden bisher stets in LISP programmiert. Einen Überblick über mögliche Anwendungsprobleme lieferten wir in der November-Ausgabe dieser Zeitschrift. Für den ST-Programmierer steht nun eine LISP-Version in der „Public-Domain Software" (Disk Nr. 7) zur Verfügung.

Woher kommt XLISP?

XLISP von David Michael Betz ist wohl die älteste Public-Domain-Version von LISP. Der Interpreter ist in C geschrieben. Da der Source Code mitgeliefert wird, läuft dieser Interpreter auf allen bekannten Computern, für die ein C-Compiler lieferbar ist. Koordinator des Projektes ist der Autor selbst, der als „Senior Editor" der angesehenen amerikanischen Zeitschrift BYTE die Entwicklung der verschiedenen Versionen in der Hand behalten möchte. Der Source Code fehlt zwar auf der PD-Diskette der ST-Computer-Redaktion, ist jedoch direkt beim Autor zu beziehen; die Adresse findet sich in der PD-Dokumentation. Ich habe außerdem zehn Dollar für die Deckung der Versandkosten beigefügt.).

XLISP steht für e**X**perimental **LISP**. Entsprechend vielzählig sind die verfügbaren Versionen. Auf der PD-Disk Nr. 7 finden sich die Versionen 1.4 und 1.5. Auf meine Anfrage nach dem Source Code wurde mir jedoch gleich die Version 1.7 zugeschickt. Die Versionen 1.4 und 1.5 unterscheiden sich durch die Einbindung von Gleitkomma-Routinen in die Version 1.5 sowie die Änderung der Routinen für die objektorientierte Programmierung. Ab Version 1.5 unterscheidet der Interpreter Groß- und Kleinschreibung nicht mehr. In der Version 1.5 steckt außerdem noch ein ohne Source Code nicht näher lokalisierbarer Fehler: Beim Laden des Prolog Interpreters "Prolog.LSP". Die Version 1.7 ist nun fast fehlerfrei. Allerdings gelingt es in dieser Version nicht, Objekte und Klassen zu definieren, da die newvector-Routine, die neuen Speicherplatz für die Objectvectoren bereit stellt, nicht richtig arbeitet. Dieser Fehler ist von mir inzwischen behoben worden: XLISP steht damit als Version 1.71 zur Verfügung. Wir beschreiben diese Version, in der wir bisher keinen Fehler entdeckt haben.

Eigenschaften

Die Dokumentation besteht aus einem 43-seitigen File auf der Diskette. Jede Version enthält leichte Änderungen. Der Autor möchte die Versionen im Laufe der Zeit immer mehr dem Common Lisp Standard anzugleichen. Aus dem Umfang der Dokumentation geht schon hervor, daß David Betz sie eher als „Quick Reference", als schnelle Informationsquelle verstanden wissen möchte. Für Anfänger empfiehlt sich daher unbedingt die Benutzung eines Lehrbuches. Wegen der inzwischen erreichten starken Kompatibilität zu Common Lisp ist das Lehrbuch von Winston und Horn [2] hervorragend geeignet, falls man vor der Sprachbarriere (das Buch ist in Englisch geschrieben) nicht zurückschreckt. Unter der deutschen Literatur ist für Anfänger am ehesten das Werk von Hamann [1] für Anfänger zu empfehlen. Nur für die objektorientierte Programmierung sollte man neuere Literatur hinzuziehen, da XLISP hier gegenüber [Hamann] deutlich abweicht und dem Konzept von Smalltalk nähergerückt ist. Aber dieses Kapitel muß man als Anfänger ja nicht sofort in Angriff nehmen.

Zum **Starten des Programms** sollte man sich unbedingt ein Menü schreiben oder ein User Shell verwenden. Ich habe das MENU+ Programm aus dem Lattice C Compiler zusammen mit einer 500 kByte RAM-Disk aus der PD installiert. Alle Programme können dann durch Anklicken aus dem MENU+ Menü aufgerufen und editiert werden. Man muß lediglich das File "MENU.INF" entsprechend Abb. 1 mit Hilfe des Editors abändern. Ein Editor ist übrigens dringend erforderlich, weil das Erstellen und Abspeichern von Funktionen direkt im Interpreter Mode von XLISP zumindest mühsam ist. Und durch die Verwendung der RAM-Disk erhält man fast schon TURBO-Komfort. Das Programm MENU+ ist sowohl im Pascal Compiler von Metacomco enthalten als auch einzeln erhältlich. Wenn es noch nicht geschehen ist, sollte XLISP als TTP-Programm vereinbart werden.

Dann nämlich werden alle in der Parameter Box angegebenen Programme beim Start automatisch geladen (.LSP wird automatisch hinzugefügt!). In Abb. 1 werden stets die Programme PP.LSP und TRACE.LSP geladen. Diese Programme sind beim Austesten von LISP-Programmen sehr hilfreich und befinden sich auf der PD-Disk Nr. 7. "PP.LSP" bringt ein pretty print (also ein gut gegliedertes Listing) einer Funktion auf den Bildschirm oder ein sonstiges Ausgabegerät. "TRACE.LSP" ermöglicht es, den Aufruf von Funktionen zu verfolgen. Soll etwa die Funktion TEST ausgedruckt werden, so geschieht das mit **(PP TEST)**. Das Einschalten des TRACE-Modus für diese Funktion erfolgt dagegen durch **(TRACE 'TEST)**. Vergißt man das Hochkomma, so wird nicht die Funktion, sondern der evaluierte Wert der Funktion in die TRACE-Liste aufgenommen. Mit **(UNTRACE 'TEST)** wird die Funktion aus der TRACE-Liste wieder herausgenommen. Man sollte sich übrigens wirklich die Mühe machen, die Beispielprogramme genau zu studieren. Man kann von ihnen viel über die Besonderheiten, etwa die Verwendung optionaler Parameter, lernen. So kann beispielsweise die PP-Funktion das Listing auch auf einem Drukker oder File ausgeben. Es muß dann zunächst ein File eröffnet werden, und zwar auf ähnliche Weise wie in C mit **(SETQ DRUCKER (OPENO "1st:"))**, wenn der Drucker zur Ausgabe eröffnet werden soll. Anschließend kann PP veranlaßt werden, den Ausdruck auf den Drucker zu leiten, indem DRUCKER als optionaler Parameter angegeben wird (z. B.: **(PP TEST DRUCKER)**). **(CLOSE DRUCKER)** schließt den File wieder. Auf die gleiche Weise können auch Files auf Diskette zum Lesen und Schreiben geöffnet werden.

Tabelle 1 faßt die in XLISP verfügbaren Funktionen zusammen. Darin sind die Funktionen, die in COMMON LISP nicht verfügbar sind, **fett** gedruckt. Bei diesen Funktionen handelt es sich meist um solche, die leicht in C zu programmieren sind und den iterativen Programmierstil unterstützen. Die Freunde puren LISPs werden hierüber die Nase rümpfen, aber manchmal können diese Funktionen ganz praktisch sein. Systemfunktionen, die in XLISP anders implementiert sind als in COMMON LISP oder

```
######################################
#                           #
# Standard C MENU+ info file     #
#                           #
# Copyright (c) 1986 Metacomco plc #
#                           #
######################################
#
# Tools menu item definitions
#
#  Form: <item name> = <command line definition>
#

TOOLS

EDIT         = {command_dir}\ED.TTP {path}\{file}.{type} {editor_opts}
LISP      = {command_dir}\XLISP.TTP {compiler_opts} {path}\{file}.{type}

#
# File menu item definitions
#
# Form: <item name> = <file pattern>
#

FILE

Choose     =  *.*
Choose LSP =  *.LSP

#
# Option menu item definitions
#
# Form: <option name> = <initial value>
#

OPTIONS

CURRENT_DIR   = C:
PATH          = C:
COMMAND_DIR   = C:
EDITOR_OPTS   =
COMPILER_OPTS = PP TRACE
```

Abb. 1: Das File MENU+

Tabelle 2: COMMON LISP Funktionen, die nicht in XLISP Version 1.7 aufgeführt sind.

(adjoin), (array-dimension), (ash), (atan), (defstruct), (first), (format), (gcd), (intersection), (isqrt), (logtest), (mapcan), (merge), (pairlis), (proclaim), (psetq), (remove-duplicates), (remove-if), (remove-if-not), (rest), (round), (second), (set-difference), (sort), (special), (step), (symbol-function), (trace), (union), (unless), (when), (y-or-n-p).

Tabelle 3: Datentypen in Xlisp

Integer, Float, Strings, Objekte, Felder, Symbole, Filepointer, Unterprogramme (eingebaute) und E/A-Ströme.

Tabelle 1: LISP Funktionen in XLISP 1.7
Funktionen, die von COMMON LISP abweichen, sind *kursiv* gedruckt. Solche, die in XLISP, aber nicht in COMMON LISP vorkommen, sind **fett** gedruckt.

Evaluationsfunktionen:
(eval), (apply), (funcall), (quote), (function), **(backquote)**, (lambda).

Symbolfunktionen:
(set), (setq), (setf), (defun), (defmacro), (gensym), **(intern), (make-symbol)**, (symbol-name), **(symbol-value), (symbol-plist), (hash).**

Property Listen Funktionen:
(get), **(putprop)**, (remprop).

Array Funktionen:
(aref), (make-array).

Listenfunktionen:
(car), (cdr), (cxxr), (cxxxr), (cxxxxr), (cons), (list), (append), (reverse), (last), (member), (assoc), *(remove)*, (length), **(nth), (nthcdr), (mapc)**, (mapcar), **(mapl), (maplist)**, (subst), (sublis).

Destruktive Listenfunktionen:
(rplaca), (rplacd), (nconc), (delete).

Prädikatsfunktionen:
(atom), **(symbolp)**, (numberp), (null), (not), (listp), **(consp)**, *(boundp)*, (minusp), (zerop), **(plusp)**, (evenp), **(oddp), (eq)**, (eql), (equal).

Kontrollstrukturen:
(cond), (and), (or), (if), **(case)**, (let), (let*), **(catch), (throw).**

Schleifenstrukturen:
(do), (do*), (dolist), (dotimes).

Programmeigenschaften:
(prog), **(prog*)**, (go), **(return)**, (prog1), (prog2), (progn).

Debugging und Fehlerbehandlung:
(error), (cerror), (break), **(clean-up) (top-level), (continue), (errset), (backtrace), (evalhook).**

Arithmetische Funktionen:
(truncate), (float), (+), (—), (*), (/), **(1+), (1—)**, (rem), (min), (max), (abs), **(random)**, (sin), (cos), **(tan)**, (expt), **(exp)**, (sqrt).

Bitweise logische Funktionen:
(logand), (logior), (logxor), (lognot).

Relationsprädikate:
(<), (<=), (=), (/=), (>=), (>).

String Funktionen:
(char), (string), (strcat), (substr).

Ein/Ausgabe Funktionen:
(read), (print), (prin1), (princ), (terpri), **(flatsize), (flatc).**

File E/A Funktionen:
(openi), (openo), (close), (read-char), (peek-char), (write-char), (read-line).

Systemfunktionen:
(load), **(transcript)**, *(gc), (expand), (alloc), (mem)*, **(type-of), (peek), (poke), (address-of)**, *exit).*

dort nicht benötigt werden, sind hell gedruckt. Die Unterschiede sind zumeist von untergeordneter Bedeutung.

Tabelle 2 zeigt die Funktionen, die zwar unter COMMON LISP, nicht aber unter XLISP verfügbar sind. Einige dieser Funktionen sind aber bereits auf der PD-Disk Nr. 7 als Lisp-Programme vorhanden (z. B. "TRACE.LSP").

Tabelle 3 faßt die in XLISP 1.7 vorhandenen Datenstrukturen zusammen. Wie man sieht, muß XLISP 1.7 den Vergleich mit teureren LISP Versionen nicht scheuen. Allerdings sind einige der Datenstrukturen in Version 1.7 brandneu hinzugekommen. Ein Beispiel ist der Datentyp Array. Eigentlich erübrigt sich wohl so etwas wie ein Benchmark-Test (einem geschenkten Barsch...), aber um eine Idee von der Leistungsfähigkeit von XLISP zu bekommen, habe ich ein rekursives Quicksort-Programm geschrieben, das ich in Bezug zu den entsprechenden Zeiten der BASIC- (iteratives Quicksort), Logo- (eingebauter Sort Befehl) und C-Version, gesetzt habe (Tabelle 4). Dabei ergab sich für 100 Elemente ein Stack Overflow Error für die rekursive Quicksort Routine in XLISP. Das muß mit einem Bug in der Stack Verwaltung zusammenhängen, denn einen 1 MByte Rechner sollte ein 100 Element-Quicksort-Programm nicht aus dem Sattel werfen. Für diese Vermutung spricht auch eine String-Fehlermeldung, falls man den zur Verfügung stehenden Speicherplatz mittels **expand** zu stark erweitert. Ich habe deshalb das in der PD-Disk Nr. 7 befindliche Sortierprogramm durch direktes Einfügen ebenfalls benutzt. Da es weniger verschwenderisch mit dem Stack umgeht, läuft die Sortierroutine (allerdings in schneckenhaftem Tempo).

Nun ja, LISP ist eben keine Compilersprache für gewöhnliche Arithmetik.

Ein großer Vorteil von XLISP ist aber die Möglichkeit, objektorientiert zu programmieren. Man kann sich dabei in einer SMALLTALK-ähnlichen Syntax versuchen. Da dieses Kapitel auch in keinem der genannten LISP-Lehrbücher in ähnlicher Weise behandelt wird, möchte ich kurz die Grundlagen unter Verwendung der XLISP-Syntax erörtern.

Objektorientierte Programmierung in XLISP

In normalen Programmiersprachen wirkt eine aktive Prozedur oder Funktion auf die passiven Daten ein. Anders in objektorientierten Sprachen. Hier ist das Objekt (Daten) zentrales Element. Will man, daß eine bestimmte Operation durchgeführt wird, übermittelt man dem Objekt eine Mitteilung, worauf die gewünschte Operation mit einer geeigneten Methode ausgeführt wird. Bei der Definition der Datenstrukturen (Objekte) wird eine strenge Hierarchie beachtet. Abb. 2 zeigt diese Hierarchie. Die **Klasse** enthält alle nötigen Informationen, um die **Instanzen** (engl.: **Instance**) zu benutzen und zu konstruieren. Die **Instanzen** einer **Klasse** sind dann die Datenstrukturen (Objekte), die die Daten (z. B. Zahlen) enthalten, mit denen wir arbeiten wollen. Jede **Instanz** gehört zu einer **Klasse**. Jede **Klasse** besitzt mehrere **Instanzen**. Jede **Klasse** erbt alle Eigenschaften der übergeordneten Superklasse. Jede **Instanz** einer **Klasse** natürlich auch. Um ein anschauliches Beispiel zu geben, habe ich das Programm **Differentiate** aus [2] in objektorientierte Schreibweise umgesetzt (Listing 1). Die ersten drei Funktionen dienen lediglich der besseren Lesbarkeit. Da die Funktionsterme in LISP-Notation eingegeben werden müssen (selbstverständlich gibt es Programme, die von der uns gewohnten Infix-Notation in die LISP-Notation (Präfix) umwandeln, aber immer sachte...), ist der car der Liste immer der Operator. Arg1 und Arg2 sind das erste und zweite Argument der Funktionsliste und stellen somit die zu verknüpfenden Funktionsterme dar. Als nächstes wird dann die Klasse **Regeln** erzeugt. Diese erbt alle Eigenschaften der obersten aller Klasse mit dem Namen **Object**. Dann wird die Instanz **Ableitungsregel** als Instanz der Klasse **Regeln** erzeugt. Diese Instanz ist nun in der Lage unsere Nachrichten und Methoden zu speichern, während die Klasse **Regeln** lediglich eine Schablone für die Instanz darstellt. Nun werden die Nachrichten für die Instanz **Ableitungsregel** der Klasse **Regeln** definiert. Es handelt sich bei den Nachrichten lediglich um die Operatoren, welche die verschiedenen Methoden auslösen. Das ist wie im richtigen Le-

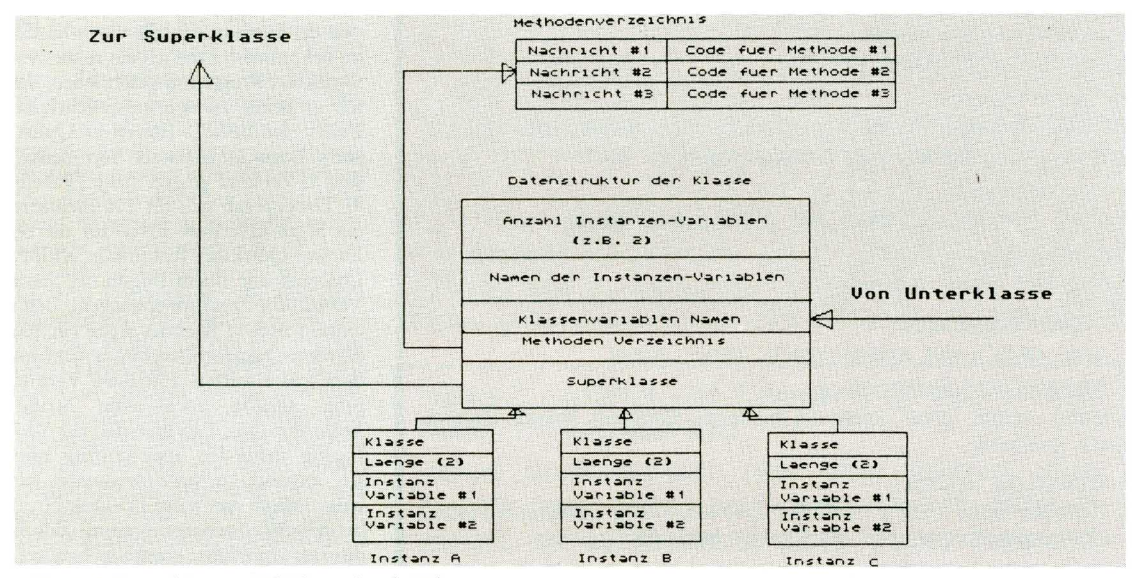


Abb. 2: Hierarchie von Objekten (nach [3])

Tabelle 4: Rechenzeiten der Quicksort Routine (in Sekunden):

Anzahl Elemente	BASIC	C	LOGO	XLISP Einfügen	XLISP Quicksort
50	—	—	—	8	5
100	6	0,03	3	20	—

```
;Listing 1.
;Objektorientierte Ableitung von Funktionen.
;Die Funktionen müssen in LISP Notation eingegeben werden.
;Beispiel f(x)=3x^2 als (* 3 (potenz x 2))
;Der aufrauf für die Ableitung dieser Funktion lautet dann:
;(ableitung '(* 3 (potenz x 2)) 'x)

(defun operator (list) (car list))

(defun arg1 (list) (cadr list))

(defun arg2 (list) (caddr list))

;Definiere die Klasse "Regeln".

(setq regeln (class :new '()))

;Erzeuge eine neue Instanz der Klasse "Regeln".

(setq ableitregel (regeln :new))

;Definiere die Nachrichten für die Operatoren und lege sie
;in der Property-Liste "opmessages" ab.

(putprop 'opmessages ':+ '+)
(putprop 'opmessages ':- '-)
(putprop 'opmessages ':* '*)
(putprop 'opmessages ':/ '/)
(putprop 'opmessages ':potenz 'potenz)

;Das eigentliche LISP Programm zur Ableitung einer Funktion f.
;f ist die Funktion, x ist die Ableitungsvariable

(defun ableitung (f x)
   (cond ((atom f) (cond   ((equal f x) 1)
                           (t 0)))
         (t (ableitregel (get 'opmessages (operator f)) f x))))

;und nun die einzelnen Regeln:

;Summenregel
(regeln :answer :+ '(f x)
      '((+ (ableitung (arg1 f) x) (ableitung (arg2 f) x))))

;Differenzregel
(regeln :answer :- '(f x)
      '((- (ableitung (arg1 f) x) (ableitung (arg2 f) x))))

;Produktregel
(regeln :answer :* '(f x)
      '(`(+ (* ,(arg1 f) ,(ableitung (arg2 f) x))
            (* ,(arg2 f) ,(ableitung (arg1 f) x)))))


;Quotientenregel
(regeln :answer :/ '(f x)
      '(`(/ (- (* ,(ableitung (arg1 f) x) ,(arg2 f))
               (* ,(arg1 f) ,(ableitung (arg2 f) x)))
            (* ,(arg2 f) ,(arg2 f)))))

;Potenzregel
(regeln :answer :potenz '(f x)
      '(`(* ,(arg2 f) (* (potenz ,(arg1 f) ,(- (arg2 f) 1))
                          ,(ableitung (arg1 f) x)))))
```

ben: durch + verknüpfte Terme werden mit Hilfe der Summenregel abgeleitet etc. Die Nachrichten sind in einer sogenannten Property-Liste abgelegt, deren Suchschlüssel der Operator und deren Wert die zugehörige Nachricht ist.

Aus dieser Property-Liste (**opmessages**) wird dann die Operatornachricht zusammen mit der Funktion **f** und der Ableitungsvariablen **x** an die Instanz **Ableitungsregel** der Klasse **Regeln** übergeben, um im Hauptprogramm die Ableitung vorzunehmen. Es sieht zwar aus, als ob die Ableitungsfunktion nicht rekursiv wäre, aber in den einzelnen Methoden kommt natürlich wieder die Ableitung vor, sodaß die Lösung auch rekursiv abläuft. Als letztes werden die einzelnen Methoden für die verschiedenen Nachrichten an das Objekt **Regeln** definiert.

Starthilfe für XLISP-Anfänger

Um LISP-Anfängern den Einstieg zu erleichtern, hier die ersten Hinweise wie XLISP zu starten ist und was der LISP-Interpreter eigentlich macht. Die ersten Schritte in XLISP unternimmt man am besten im direkten Interpretermodus. XLISP ist als Interpretersprache den meisten BASIC Anwendern von der Arbeitsweise vertraut. Man gibt einen Befehl ein und erhält Die Schreibweise ist etwas ungewohnt, denn jeder LISP-Befehl ist in runde Klammern einzuschließen. Um das System zu starten, klicken wir "XLISP. TTP" an. In der Dialogbox geben wir "PP" und "TRACE" an. XLISP startet nun und zeigt die Copyright-Mitteilung. Dann lädt das Programm "PP. LSP" und "TRACE.LSP". Mit diesen beiden Funktionen haben wir die oben erwähnten Utilities stets griffbereit. Danach zeigt der Interpreter mit ">" an, daß er bereit ist, LISP Funktionen zu evaluieren (d. h. den Wert der Variablen X auszugeben). Tippt man (X) ein, antwortet der Interpreter mit der Fehlermeldung "Unbound Variable". Logisch: Der Interpreter versucht den Wert der Variablen X zu bestimmen und erkennt, daß diese noch nicht an einen Wert gebunden wurde. Weist man der Variablen mit **(SETQ X 3)** den Wert 3 zu, dann ergibt dieselbe

Eingabe **(X)** die Antwort 3. Wegen der Eigenschaft, nach der Eingabe sofort den Wert der Funktion oder der Variablen zu ermitteln, zählt XLISP zu den EVALLISP-Interpretern. Die Evaluierung kann auch unterbunden werden mit der Funktion QUOTE, abgekürzt '. Die Anweisung **('X)** gibt die Antwort X, weil durch ' die Evaluierung unterbunden wurde. Da wir schon bei den Variablen sind: LISP kennt als Datenstrukturen eigentlich nur „Atome" (nicht weiter zerlegbare Datenstrukturen) und „Listen" (von ihnen hat LISP seinen Namen: **LIS**t **P**rocessing). Man kann sich Atome in Lisp als wortähnliche und Listen als satzähnliche Gebilde vorstellen. Atome und Listen werden gemeinsam symbolische Ausdrücke (engl.: symbolic Expressions: SEXPR) genannt. Das obige kleine Beispiel zeigt schon die Fähigkeit von LISP, mit Symbolen umzugehen. Schauen wir uns zunächst einige Lisp-Atome an: Alle Zahlen sind Lisp-Atome (numerische Atome). Bei der Bearbeitung dieser Atome wählt Lisp die Präfix Notation, d. h. die zu bearbeitenden numerischen Atome (Zahlen) werden zusammen mit einem vorgezogenen Operatorsymbol (einem symbolischen Atom) in einer Liste zusammengefaßt. Zum Beispiel ergibt die Eingabe **(+ 2 6)** das Ergebnis 8 auf dem Bildschirm. Die einzelnen zu bearbeitenden Symbole (das symbolische Atom und die beiden numerischen Atome) wurden hierzu in runden Klammern geschrieben und damit zu einer Liste zusammengefaßt. Und was ergibt **(+ 2 X)**? Falls Sie das erste Beispiel eingetippt haben, natürlich 5. Klar?

Richtig, das symbolische Atom (kurz: Symbol) X wurde mit setq an das numerische Atom 3 gebunden. Eine **Liste** besteht also aus einer linken sich öffnenden Klammer gefolgt von 0 oder mehr Atomen oder Listen, gefolgt von einer rechten sich schließenden Klammer. Die Liste 6+ (* 2 3) (/ 1 2)) ist also nichts anders als die Präfix-Schreibweise des einfachen algebraischen Terms ((2 * 3) + (1/2)), welcher den Wert 6.5 besitzt. Die Hierarchie der verschiedenen Datentypen in LISP zeigt Abb. 2. Dieses Diagramm gilt für XLISP erst von der Version 1.5 an, da bis dahin die Gleitkommazahlen fehlen. Aber für numerische Rechnungen allein ist ein LISP-Interpreter ohnehin zu schade. Das erledigen die altbekannten PASCAL- oder C-Compiler schneller und (da XLISP in C geschrieben wurde) mit gleicher Genauigkeit (siehe obigen "Benchmark"-Test).

Schwerpunkt von LISP ist, wie der Name sagt, die Listenverarbeitung. Deshalb schauen wir uns in dieser kurzen Einführung wenigstens noch die wichtigsten Funktionen zur Listenverarbeitung an. Diese sind CAR und CDR (sprich kaar und kjuudr). Die Bezeichnung beider Funktionen ist rein historisch und hat keinen symbolischen Wert mehr. Man übersetzt die Funktionen wohl am besten mit ERSTES und REST. CAR liefert nämlich das erste Objekt (Atom oder Liste) einer Liste, während CDR eine Liste zurückgibt, in der alle Objekte der Ausgangsliste enthalten sind, bis auf das erste. Nehmen wir folgendes Beispiel: Wir wollen CAR und CDR der Liste (A B C) erhalten. Dazu geben wir ein **(CAR '(A B C))** und erhalten A. Nach **(CDR ' (A B C))** erfolgt die Ausgabe (B C).

Beachten Sie das Hochkomma vor (A B C). Seine Bedeutung wird klarer, wenn man zunächst das Atom LISTE an die Liste (A B C) bindet: **(SETQ LISTE '(A B C))**. **(CAR LISTE)** (ohne Hochkomma) liefert jetzt den gleichen Wert (nämlich A), weil LISP zunächst den Wert von LISTE evaluiert, (A B C) erhält und davon den CAR ermittelt. In (CAR (A B C)) würde LISP auch versuchen, den Wert von (A B C) zu evaluieren. Mangels Bindung würde LISP jetzt allerdings einen Fehler melden. Das Hochkomma stoppt also die Evaluierung und übernimmt den nachfolgenden Ausdruck unevaluiert. **(CAR 'LISTE)** würde ebenfalls zu einem Fehler führen, weil LISTE unevaluiert ein Atom und keine Liste ist. Diese essentiellen LISP-Befehle sind in XLISP identisch mit der in [2] gegebenen Nomenklatur. Abb. 3 zeigt nochmal den Evaluierungsprozeß in der Übersicht.

Natürlich reicht hier der Platz für eine ausführliche Einführung in LISP mit Hilfe des XLISP-Systems nicht aus. Ich hoffe jedoch, daß der ungeübte Benutzer in die Lage versetzt wurde, XLISP zu starten und anhand eines der aufgeführten Lehrbücher tiefer in diese großartige Symbolmanipulationssprache einzudringen. Es lohnt sich in jedem Fall, das XLISP System aus der PD zu besorgen und die einschlägigen Lehrbücher und mitgelieferten Beispielprogramme durchzuarbeiten.

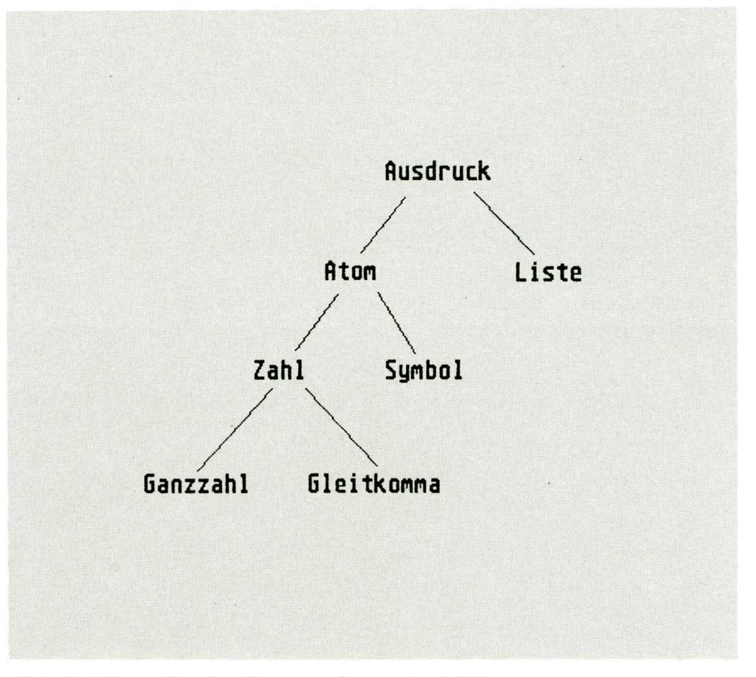


Abb. 3: Hierarchie der Datenstrukturen in LISP

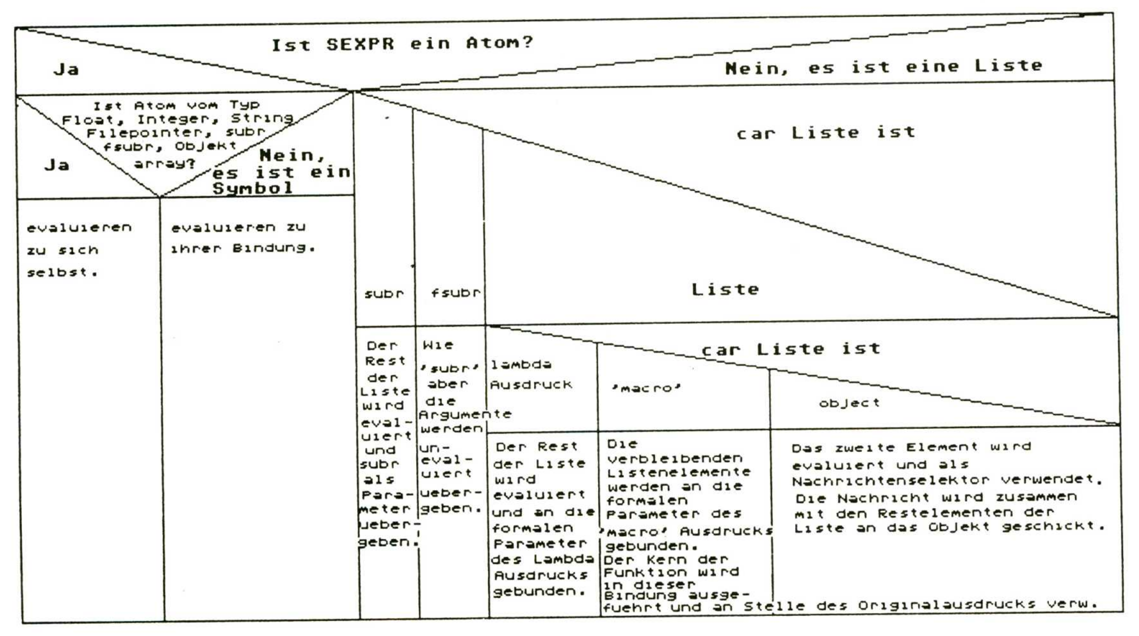


Abb. 4: Struktogramm des Evaluationsprozesses

Kleines Lexikon für objektorientierte Programmierung in XLISP

Instanz
Hier wird die im Objekt Klasse definierte Datenstruktur realisiert, und die aktuellen Werte der Instanz-Variablen werden abgelegt. Alle Instanzen einer Klasse haben die gleiche (in der Klasse festgelegte) Datenstruktur.

Instanzen-Variable
Private Datenstruktur, die nur dem Objekt und seinen Methoden zugänglich ist. Hier werden Informationen über die verschiedenen Instanzen der gleichen Klasse gespeichert.

Klasse
Datenstruktur, die die Informationen für alle gleichartigen Objekte enthält. Siehe Abb. 4.

Methode
Lisp-Programm, das auf eine Nachricht hin aktiviert wird und nur für ein bestimmtes Objekt definiert ist.

Nachricht
Sie veranlaßt ein Objekt, den vorher definierten Code auszuführen.

Objekt
Zentrale Datenstruktur, die Speicherplätze für private Informationen und einen Satz von Methoden enthält.

Literaturangabe:

[1] Hamann, C. M. Einführung in das Programmieren in LISP. Walter de Gruyter. Berlin 1982.

[2] Winston, Patrick Henry und Berthold Klaus Paul Horn. LISP. Second Edition. Addison Wesley. Reading 1984.

[3] Pascoe, Geoffrey A. Elements of Object-oriented Programming. BYTE August 1986. p. 139 ff.

Dr. Karl Sarnow

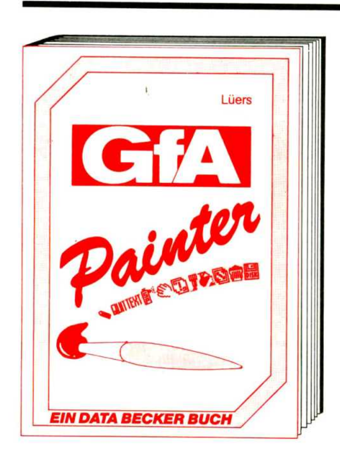
Alles über





Das neue GfA-BASIC bietet Ihnen mit seinem mächtigen Befehlssatz alle Möglichkeiten zur echten Systemprogrammierung des ATARI ST. Um wirklich alle Features nutzen zu können, bietet DATA BECKER ein völlig neues Konzept: Geboten wird nicht nur eine nackte Befehlsübersicht, mit einer Vielzahl von Syntax- und Parametererklärungen, sondern ein komplettes, ausbaufähiges Graphic Construction Set, das ein echtes „Learning by Doing" ermöglicht. Schrittweise und didaktisch perfekt werden einzelne Befehle in das Gesamtprogramm integriert. So haben Sie am Ende des Buches nicht nur einzelne Beispielroutinen, sondern das fertige Super-Grafik-Programm, das alle Standards enthält und zusätzlich über folgende Funktionen verfügt: ● Echter Trickfilm-Modus ● Icon- und Pattern-Editor ● 3-D-Animationsgrafik ● Clipping ● Patch-Grafiken, incl. Self-Option u. v. m. Nebenbei gibt der Autor eine Menge nützlicher Tips & Tricks wie z. B. Rastertechnik, Sprites, System-Calls, Window- und Objectprogrammierung und GEM-Procedures. Insgesamt also brandneues Detailwissen in kompakter Buchform. Ein echtes DATA BECKER Superbuch von einem wirklichen BASIC-Profi!

Hardcover, 468 Seiten, DM 49,–



Lassen Sie sich von einem DATA BECKER Profi zeigen, wie man Spitzenprogramme in GfA-BASIC schreibt. Das in diesem Buch dokumentierte Super-Malprogramm enthält Features, die bisher nicht für möglich gehalten wurden. Ohne schwer begreifliche Assemblerroutinen eintippen zu müssen, bekommen Sie hier Know-how pur. Alles natürlich erläutert und mit vielen Ideen für eigene Erweiterungen. Hier ein Ausschnitt aus den über 100 Features: ● Blöcke markieren, vergrößern, verkleinern, drehen, spiegeln und invertieren ● Malen mit beliebigen Bildschirmausschnitten ● automatische Anpassung an verfügbaren Speicherplatz ● Spraydose mit 4 Strichstärken ● 6 verschiedene Strichstärken und Stricharten ● X-Ray-Line-Modus ● Spezielle Vertikal- und Horizontal-Line-Funktion ● 5 Polygon-/Polymarker-Darstellungen ● Volles DIN A 4-Arbeitsblatt ausdruckbar, 2. Grafikseite für Bibliotheken nutzbar ● Optionales Koordinatennetz ● Mustereditor und Malen mit selbsterstellten Mustern ● 36 Standardmuster ● Zoom-Modus ● UNDO-Funktion bis zu vier Schritte rückwärts. Super-Features im Textmodus: ● variabler Zeichen- und Zeilenabstand ● Schreiben in 5 Richtungen incl. Vertikal-Schrift für Tabellen ● vielfältig kombinierbare Schriftarten und -stile. Mit diesem Buch bekommen Sie perfektes DATA BECKER Know-how! Inkl. Anforderungsschein für Programmdiskette.

386 Seiten, DM 39,–



Daß das GfA-BASIC eine der leistungsfähigsten BASIC-Versionen ist, die es für den ST gibt, hat sich mittlerweile herumgesprochen. Ein mächtiger Befehlssatz sorgt dafür, daß auch in einer Sprache wie BASIC fast kein Problem mehr ungelöst bleibt. Bis auf das eine: Sich die Syntax jedes Befehls zu merken. Aber dafür gibt es ja den DATA BECKER Führer zum GfA-BASIC. Eine komplette Befehlsübersicht – natürlich auch die neuen Befehle des GfA-BASIC 2.0 – inklusive aller nötigen Parameter und Syntax-Erläuterungen sorgt dafür, daß Sie jeden Syntax Error sofort überprüfen können. Auf diese Weise haben Sie die Chance, sich voll und ganz auf das Wichtigste zu konzentrieren: auf Ihr Programm.

Der DATA BECKER Führer zum GfA-BASIC, 254 Seiten, erscheint ca. 11/86, DM 24,80

BESTELL-COUPON

Einsenden an: DATA BECKER · Merowingerstr. 30 · 4000 Düsseldorf 1

Bitte senden Sie mir:

zzgl. DM 5,– Versandkosten

☐ per Nachnahme ☐ Verrechnungsscheck liegt bei

Name

Straße

Ort

ST 17

DATA BECKER

Merowingerstr. 30 · 4000 Düsseldorf · Tel. (0211) 310010

CAMBRIDGE LISP – ATARI ST

KI auf dem ST

Während das Stichwort Künstliche Intelligenz (KI) in der Computerszene seit einigen Monaten hoch im Kurs steht, war es lange Zeit im Zusammenhang mit dem ATARI ST kein Thema. Ein Anachronismus, der nicht zuletzt durch **METACOMCOs Cambridge Lisp** als aufgehoben zu betrachten ist.

Dies ist umso mehr zu begrüßen, als der ATARI ST für diese an Arbeitsgeschwindigkeit und Speicherbedarf anspruchsvollen Produkte die besten Voraussetzungen mitbringt.

Cambridge Lisp verfügt zudem über einige Features, die man selbst bei weit teuereren Lisp-Systemen vergeblich sucht – als besonderen Leckerbissen einen **integrierten Compiler.**

Lieferumfang	
Lisp-Programm	160 KB unter TOS laufendes Hauptprogramm
Image-Ordner	252 KB Module wie Compiler und GEM-Schnittstelle
GEM-Bibliothek	140 KB 19 kommentierte Quelldateien
Beispielprogramme	25 KB u. a. GEM-Windows, „Expertensystem"
Bildschirm-Editor	32 KB extern und in Lisp integriert
Menu+, Util	34 KB GEM-Shell mit Hilfsprogramm
Dokumentation:	> 300 Seiten (englischsprachig)
Speicherbedarf:	500 KB (empfehlenswert: ≥ 1 MB)
Adreßraum:	16 MB
Preis:	490,– DM

Lieferumfang und Dokumentation

Das komplette Programmpaket ist auf zwei Disketten verteilt. Auf der ersten befinden sich das Haupt-Programm, eine GEM-Bibliothek und einige Beispielprogramme. Auf der zweiten ein „Image"-Ordner, in dem einige unverzichtbare, zur „imaginären Lisp-Welt" gehörige Programm-Module enthalten sind. Sie enthält außerdem weitere, von anderen METACOMCO-Produkten bekannte, Programme wie einen unter TOS laufenden Bildschirm-Editor und die GEM-Shell menu+.

Menu+ ist ein weitgehend frei konfigurierbares Hilfsprogramm zum Aufruf anderer Programme. Es erlaubt u. a. eine komfortable Übergabe optionaler Parameter mit Angaben über den von Lisp zu nutzenden oder freizulassenden Speicherplatz, über die aktuelle Ausgabedatei und dergleichen mehr.

Die englischsprachige Dokumentation von weit über 300 Seiten kann sich sehen lassen. Nach der Darstellung der GEM-Shell und des Editors werden in etwas über 80 Seiten die wichtigsten Features von **Cambridge Lisp** in thematischer Reihenfolge beschrieben. Hieran schließt sich ein 100 Seiten umfassender, bis auf die GEM-Routinen vollständiger Referenzteil an. Den AES- und VDI-Funktionen, die vollständig und mit ihren Standard-Namen implementiert sind, werden 70 Seiten gewidmet. Hier wäre ein zusätzliches Kapitel zur GEM-Programmierung wünschenswert. In verschiedenen Anhängen erhält der Leser weitere Informationen zum Aufruf von BIOS-, XBIOS- und GEMDOS-Befehlen und zu Fragen der Kompatibilität. Schließlich findet man einige hilfreiche Beispielprogramme und einen 19seitigen Index angefügt.

Sprachumfang

Cambridge Lisp wurde ursprünglich für eine Großrechneranlage entwickelt. Im wesentlichen handelt es sich dabei um eine Erweiterung von **Standard Lisp**, wie es von Marti u. a. formuliert wurde (SIGPLAN Notices 14(10), 1979).

Der implementierte Sprachumfang kann sich sehen lassen: die Objektliste zählt 727 Funktionen der insgesamt 885 implementierten Lisp-Objekte. Bei einem nicht unbeträchtlichen Teil davon handelt es sich jedoch um Hilfsfunktionen des Compilers. Weiterhin ist zu berücksichtigen, daß dabei die GEM-Funktionen mitgerechnet sind und viele Funktionen in mehr oder minder effizienten und restriktiven Varianten existieren.

Angesichts dieser Fülle verbietet sich der Versuch, einen möglichst vollständigen Überblick über den implementierten Sprachumfang zu geben. Unverzichtbar ist die Darstellung der verfügbaren Funktions- und Datentypen. Bevor wir uns schließlich dem Compiler zuwenden, werden wir noch die Bestandteile des Programmpakets betrachten, die aus einer Programmiersprache ein Programmiersystem machen: Editor, Fehlerbehandlung und ein besonderes Feature von **Cambridge Lisp** – das Module-System.

Funktionstypen und Kontrollfunktionen

Der Hinweis auf die Abstammung von **Standard Lisp** sollte nicht darüber hinwegtäuschen, daß es für Lisp keinen Standard gibt, auch wenn gegenwärtig mit **Common Lisp** versucht wird, einen Standard durchzusetzen. Wer an **Common Lisp** gewöhnt ist, wird die Spezialformen do und let sowie die Möglichkeit vermissen, &rest-, &aux-, &key- und &optional-Parameter anzugeben. **Cambridge Lisp** weist hier vielmehr die gleichen Features wie **Interlisp** auf, einem der auf Großrechnern verbreitetsten Lisp-Dialekte.

In beiden Dialekten werden fehlende aktuelle Parameter automatisch mit nil in-

itialisiert; Funktionen, gleich welchen Typs, können wahlweise als spread (mit einer festen Anzahl von Parametern) oder nospread (mit einer beliebigen Menge von Parametern, deren Werte in Form einer Liste gebunden werden) definiert werden.
Das Fehlen der von **Common Lisp** bekannten Spezialform „do", sowie verwandter „Funktionen" zur Bildung strukturierter Schleifen, erscheint demgegenüber als größeres Manko. Ein Manko, das auch durch den Hinweis, rekursive Funktionsaufrufe seien in Lisp die „natürlichen" Kontrollstrukturen, nicht wegzudiskutieren ist. Die von **Cambridge Lisp** zur Verfügung gestellten Makros (for, while, repeat...) sind demgegenüber weniger mächtig und flexibel. Da jedoch die Möglichkeit besteht, **Makros** selbst zu definieren, kann dies nicht so ins Gewicht fallen: der Benutzer kann solche Spezialformen mit Hilfe des Programm-Features (prog, progn, prog2, go, casego, return) und cond nach seinem eigenen Gutdünken implementieren. Erwähnenswert ist in diesem Zusammenhang noch, daß nichtlokale Ausgänge von Funktionen durch „catch" und „throw" unterstützt werden.
Einen anderen Typ von Spezialformen stellen noeval-Funktionen. Ihre Argumente werden, wie die von Makros, beim Einlesen nicht evaluiert. Im Gegensatz zu ihnen führen sie jedoch auch keine zweite Evaluation durch.

Funktionstypen

	spread		nospread	
	interpretiert	kompiliert	interpretiert	kompiliert
Eval	expr	subr	lexpr★	lsubr★
Noeval	fexpr	fsubr		
Macros★★★	macro mmacro★★			

★ Cambridge Lisp sieht keine Bezeichnungen für nospread-Funktionen vor. Der Sache nach bestehen alle Kombinations-Möglichkeiten.
★★ Maclisp-artige Notation von Makros: Macronamen als Parameter.
★★★ Hiervon zu unterscheiden sind die ebenfalls unterstützten Readmacros.

Datentypen und Datenstrukturen

Identifiers	u. a. Funktions- und Variablennamen (vgl. literale Atome)	Zusammengesetzte Listenstrukturen
Strings	"dies ist ein "string""	
Listen/dotted pairs	(a . (b . nil)) = (a b)	Assoziationslisten Propertylisten AVL-Listen
Vektoren	[34,(beliebige),'[objekte])	
Zahlen		
Ganze Zahlen	smallintegers: −2 24 bis 2 24; bignums ≥ 10 9;	
Gleitpunktzahlen	(fix 3.487529e6) = 3487529	GEM-unterstützende Typen/Strukturen
rationale Zahlen	(sqrt (rational 4 9)) = (2/3); (rational 3.5) = (7/2) (float (rational −7 2)) = −3.5	cstring, cstruktur, buffer, memory definition block
Konstanten	(make!-constant 'a 3); nil ...	
Binärfunktionen	kompilierte Lisp-Funktionen	
Special identifiers	trace:, lambda, lambdaq ...	

Datentypen und Datenstrukturen

Neben den in allen Lisp-Dialekten üblichen Assoziations- und Propertylisten kennt **Cambridge Lisp** einen weiteren Typ zusammengesetzter Listenstrukturen: **AVL**-Listen. AVL bezeichnet dabei einen Algorithmus zum Ausbalancieren von Baum-Strukturen und ist ein Akronym für die sowjetischen Wissenschaftler G. M. Adel'son, Vel'skii und E. M. Landis. Unter Verwendung der hierfür implementierten Funktionen wird automatisch dafür Sorge getragen, daß die Anzahl der Knoten von jedem Ast nicht um mehr als ± 1 divergieren.

Vektoren spielten traditionell in Lisp gegenüber den flexibleren Listen keine Rolle. Da sie jedoch durch die feste sequentielle Anordnung ihrer Elemente effizienter als über Pointer vermittelte Listenstrukturen sind, werden sie in modernere Dialekte wie **Cambridge Lisp** als originäre Datentypen aufgenommen. Als Elemente von Vektoren sind beliebige Lisp-Objekte, mithin auch Vektoren, zugelassen.

Neben ganzen Zahlen, die von beliebiger Größe sein können, und Gleitpunktzahlen kennt **Cambridge Lisp** noch rationale Zahlen, die in Form einer Liste repräsentiert werden.

Die arithmetischen Funktionen schließen Bitoperationen ebenso ein wie trigonometrische Funktionen und Berechnungen mit auf bestimmte Intervalle reduzierten Zahlen.

Ein- und Ausgabe

Das in **Cambridge Lisp** realisierte Konzept der „Datenströme" erlaubt, verschiedene Medien wie Terminal, Bildschirm oder Drucker als aktuelle Ein- oder Ausgabekanäle zu spezifizieren, um den Datenfluß zu lenken. So kann der Benutzer, beispielsweise mit dem Aufruf von Lisp, den Drucker zum aktuellen Ausgabekanal deklarieren und sich ein Ablaufprotokoll ausdrucken lassen.

Bemerkenswert ist in diesem Zusammenhang die Flexibilität und Offenheit der Zeichen-Syntax. Es besteht freier Zugang zur Syntax-Tabelle. Ohne Umstände können so die Eigenschaften von Zeichen verändert werden: ob ein Zeichen etwa als Trennzeichen oder als escape-Zeichen wirkt oder ob es als **Readmacro** fungiert. Durch solche Readmacros – die bekanntesten der bereits implementierten sind das Quote- (') und Backquote-Zeichen (') – können bereits in der Einlese-Phase bestimmte Textersetzungen vorgenommen werden.

Bestandteile des Programmiersystems	
Editor	integrierter Bildschirmeditor unter TOS; manövriert auf Lisp-Klammern; Blockoperationen, Such- und Ersetzungsfunktionen; Schnittstelle: ausschließlich Dateien, keine Funktionen
Fehler-Behandlung	über 260 implementierte Fehlermeldungen; Backtrace mit einstellbarer Informationsfülle; error, errorset (zur Kontrolle fehlerhafter Programmabläufe); trace, tracesetq, tracecount und embed – mit Ausnahme von tracesetq auch auf kompilierte Funktionen anzuwenden; kein Break-Modus
Module-System	Einrichten verschiedener Lisp-Welten (core image dumping) in Modulen abgelegte Funktionen werden bei Aufruf automatisch geladen, sofern sie nicht im Speicher sind (load on call)
Sonstiges	Automatische Speicherverwaltung (dem Bedarf anpassende Aufteilung) Prettyprinter mit verschiedenen Optionen: quote als Makro-Zeichen, evaluierbare Ausgabe (mit escape-Zeichen) u.v.a.m.

Editor

Bei dem im Programmpaket enthaltenen Editor handelt es sich um einen unter TOS laufenden Bildschirm-Editor. Er ist in zwei Versionen implementiert: die erste ist vom Desktop oder der GEM-Shell aufrufbar.

Die zweite ist in Lisp integriert, d. h. sie wird durch einen Lisp-Befehl aufgerufen und gibt nach Beendigung die Kontrolle wieder an den Lisp-Interpreter zurück. Unterstützt werden eine Reihe von Editierfunktionen, einschließlich Blockoperationen sowie Such- und Ersetzungsfunktionen. Zudem erlaubt die in Lisp integrierte Version auf Lisp-Ausdrücken zu manövrieren. Über Funktionstasten ist es möglich, den Cursor zur jeweils zugehörigen Klammer einer Liste und zur nächsten oder vorausgehenden Liste zu bewegen. Ein zweifellos für die Lisp-Programmierung nützliches Feature.

Neben der fehlenden GEM-Unterstützung ist zu bemängeln, daß der Zugang zum Editor ausschließlich über Dateien erfolgt. Zu einer komfortablen Entwicklungsumgebung gehört sicherlich auch die Möglichkeit, dem Editor Funktionen zu übergeben. In dieser Hinsicht hat beispielsweise der Editor von **DR. LOGO** mehr zu bieten, ganz zu schweigen von Lisp-Editoren, die eine Evaluation ermöglichen.

Fehlerbehandlung

Einen wesentlichen Bestandteil eines Programmiersystems stellen die Werkzeuge zur Diagnose und Kontrolle von Fehlern dar.

In **Cambridge Lisp** sind dazu eine Reihe nützlicher Funktionen vorgesehen. Mit error lassen sich benutzerdefinierte Fehlermeldungen einbauen, mit errorset Fehler abfangen und die Kontrolle über fehlerhafte Programmabläufe sichern. Mit trace kann der Benutzer die aktuellen Werte der Parameter der angegebenen Funktionen sowie deren Funktionswerte verfolgen, mit tracesetq die Wertbindungen der freien Variablen. Tracecount erlaubt, eine bestimmte Anzahl von Trace-Meldungen zu unterdrücken. Zusätzliche Informationen über die jeweils aktuelle Funktionsumgebung können mit embed erhalten werden. Diese Funktion ermöglicht reversible Modifikationen von Funktionen: der alte Funktionskörper wird bis auf Widerruf in einen neuen „eingebettet". Schließlich erfolgt nach Fehlern ein Backtrace, dessen Informationsfülle vom Benutzer eingestellt werden kann. Einziges Manko dieser sonst vorbildlichen Fehlerbehandlung ist das Fehlen eines Break-Modus, der nach einem Fehler die Untersuchung der aktuellen Umgebung und die Fortführung des Programmablaufs bei korrigierbaren Fehlern ermöglicht.

Das Module-System

Mit dem Module-Sytem weist **Cambridge Lisp** ein äußerst nützliches Feature auf, das von seiner Großrechner-Herkunft zeugt.

Das Module-System erlaubt nämlich, verschiedene „Lisp-Welten" mit eigenem Sprachumfang und eigener Syntax zu schaffen. Daß sich dies nicht nur als nützlich erweist, wenn mehrere Benutzer einer Anlage mit ihrer persönlichen „Lisp-Welt" arbeiten möchten, braucht wohl nicht weiter erläutert zu werden. Es bietet jedoch noch mehr Vorteile und ist es wert, seine Funktionsweise genauer darzustellen.

Von den im Lieferumfang enthaltenen Modulen hat eines einen besonderen Status – das Modul Lisproot. In dieser „Wurzel" ist die Umgebung festgelegt, in der sich der Benutzer nach dem Laden des Lisp-Programms findet. Mit preserve können in ihr neue Funktionen und Lisp-Objekte ebenso bewahrt werden wie Modifikationen der implementierten. Es ist sogar möglich, eine Funktion anzugeben, mit der das System nach dem Laden startet. Weniger angenehm ist, daß Lisp automatisch verlassen wird, nachdem mit preserve ein neues „core image" geschaffen wurde.

Das Module-System enthält ferner ein nützliches Instrumentarium, um mit den bei größeren Programmiersystemen auftretenden Speicherplatz-Problemen umzugehen. Praktisch machen sich solche Probleme bei Lisp oft durch erheblich reduzierte Ablaufgeschwindigkeiten bemerkbar. (Eine Erfahrung, die jeder auch mit einem kleinen Programm wie **XLISP** machen kann, wenn er nicht mit expand genügend Speicherplatz reserviert.)

Ruft man die Funktion module auf, so werden bis auf Widerruf die im folgenden kompilierten Funktionen im Binärcode abgespeichert. Gleichzeitig werden sie in die Objektliste mit einem speziellen Wert aufgenommen. Sofern solcherart gespeicherte Funktionen sich nicht im Speicher befinden, bewirkt ihr Aufruf, daß das zugehörige Modul geladen wird, bevor sie ausgewertet werden. Die Module können auch wieder aus der aktuellen „Lisp-Welt" verbannt werden. Wird der Speicherplatz zu eng, übernimmt diese Aufgabe der garbage collector. Für das Nachladen solcher Module ist eigens eine fast-load-Funktion eingebaut. Arbeitet man mit einer Ramdisk, so benötigt das Laden des

knapp 70 KB großen Compiler-Moduls ca. 2 Sekunden.

„Integrierter" Compiler für „Inkrementelle" Kompilation

Lisp-Compiler stellen immer noch eine Rarität unter den vielen Lisp-Implementationen dar – gerade auf Mikrocomputern. Ein Umstand, der sicherlich nicht wenig zu dem Haupteinwand gegen diese mächtige Sprache, sie sei zu langsam, beigetragen hat.

Die Praxis, Lisp vorrangig als Interpreter zu implementieren, hängt wohl nicht zuletzt mit ihrem Verwendungszweck zusammen: vom Standpunkt der experimentellen Erforschung „künstlicher Intelligenz" ist die effiziente, interaktive Programmentwicklung wichtiger als die Ablaufgeschwindigkeit eines bereits entwickelten Programms.

Die Kommerzialisierung der KI-Produkte mag ein Beweggrund dafür gewesen sein, dies zu verändern. Wenn aber die Arbeit mit Compilern auf eigens für Lisp konstruierte Maschinen, sogenannten Lisp-Maschinen, zur gängigen Praxis geworden ist, so liegt das wohl eher daran, daß diese Compiler mit der in der "Lisp-Gemeinde" bevorzugten Programmiermethodik des „strukturierten Wachstums" nicht im Widerspruch stehen.

Was das heißt, wird verständlicher, wenn wir die Arbeit mit dem **Cambridge Lisp Compiler** betrachten.

Der erste Unterschied zu Verfahrensweisen in anderen Sprachen besteht darin, daß der Compiler von Lisp aufgerufen wird. Dies kann explizit geschehen, indem der Funktion compile eine Liste mit den zu kompilierenden Funktionen übergeben wird. Es kann aber auch automatisch geschehen, indem der Systemvariablen ★comp ein Wert ungleich nil zugewiesen wird. Im letzten Fall werden die Funktionsdefinitionen, sobald sie dem System übergeben werden, unmittelbar kompiliert. Das geht bei den in Lisp üblichen Funktionsgrößen so schnell, daß man den Unterschied zum Interpretermodus kaum bemerkt. Und wenn man will, kann man – wiederum durch Setzen einer Systemvariablen – dafür Sorge tragen lassen, daß der „Lisp-Quell-Code" gesichert wird.

Die Arbeit mit dem Compiler gestaltet sich insofern kaum anders als die mit dem Interpreter: einzelne Funktionen können geschrieben, getestet, beliebig von anderen Funktionen aufgerufen und kompiliert werden. All dies geschieht, ohne daß sie etwa von einem „Hauptprogramm" aufgerufen werden müßten, oder daß ein externer Linkvorgang eingeleitet werden müßte. Ganz zu schweigen von der Notwendigkeit, ein Batch-Programm schreiben zu müssen, das den Quellcode durch den Compiler leitet.

Die kompilierten Funktionen können dabei nach Belieben mit zu interpretierenden gemischt werden. Ob kompiliert oder nicht, jede definierte Funktion erweitert den Sprachumfang der aktuellen „Lisp-Welt" und kann genauso benutzt werden wie eine Systemfunktion. Insofern bleibt das Konzept der „Spracherweiterung" voll erhalten.

Semantische Besonderheiten des Compilers

Einschränkend muß jedoch darauf hingewiesen werden, daß der Compiler einige semantische Besonderheiten aufweist.

Abgesehen von dem Unikum, daß das kompilierte prog2 sich wie progn verhält, sind folgende Unterschiede zu beachten.

Zum einen müssen freie Variablen als fluid deklariert sein. Dies geschieht automatisch, wenn ihnen auf Toplevel ein Wert zugewiesen wird, nicht jedoch wenn die Wertzuweisung lediglich innerhalb eines Funktionskörpers erfolgt.

Weitere Besonderheiten treten im Zusammenhang mit Spezialformen auf, den lambdaq-Ausdrücken und den Makros. Werden Funktionen kompiliert, die noch nicht definierte Makros aufrufen, so muß dies dem Compiler mitgeteilt werden. Insbesondere muß darauf geachtet werden, daß Makros während der Kompilation expandiert werden. Hat man beispielsweise eine Funktion kompiliert, in der ein Makro aufgerufen wird, und ändert man anschließend die Definition dieses Makros, so hat diese Änderung keinen Einfluß auf das Verhalten der kompilierten Funktion. Die Merkwürdigkeit, daß Funktionen mit Makros effizienter sein können als mit „normalen" Lisp-Funktionen, ist wohl darauf zurückzuführen, daß aufgrund der Makroexpansion Funktionsaufrufe unterbleiben.

Zwar unterstützt **Cambridge Lisp** die Möglichkeit, auch „normale" Funktionen ohne Funktionsaufruf zu kodieren, die zu dieser inline-Kodierung notwendigen internen Kenntnisse werden jedoch vom Handbuch nicht vermittelt.

Benchmarks

Wenn auch bei einer Sprache wie Lisp die Frage der maschinellen Effizienz gegenüber der kognitiven, die Frage der Ablaufgeschwindigkeit gegenüber der des Instrumentariums rationeller und komfortabler Programmierung „künst-

Benchmarks

Programm	n	XLISP	Cambridge Lisp	
			interpretiert	kompiliert
Fakultät	100	–	1.75"	0.56"
	500	–	16.81"	11"
	1000	–	60.47"	49"
Q	10	2.84"	5.60"	–
	12	7.85"	15.54"	–
	14	22.03"	41.94"	0.78"
Queen	4	2.47"	4.16"	–
	5	10.75"	17.62"	–
	6	49.50"	77"	2.72"
ATN-Parser	1	0.81"	3.20"/ 1.53 "	–
(n Beispielsätze)	10	5.88"	29.22"/13.13"	–
	30	18.97"	86.60"/38.53"	0.81"

licher Intelligenz" an Gewicht verliert, so ist sie dennoch von nicht unerheblichem Interesse.

Die in der Tabelle abgedruckten Ergebnisse eines Vergleichs zwischen dem Public-Domain-Programm **XLISP** und **Cambridge Lisp** bedürfen einiger Kommentare. Es sollte grundsätzlich klar sein, daß hier Unterschiedliches auf Gleiches reduziert wird. Man muß davon ausgehen, daß **Cambridge Lisp** bei der Evaluation einiges mehr zu tun hat als **XLISP**. Daher sollte es nicht verwundern, daß dieser Interpreter „langsamer" ist. Ein kaum lösbares Problem stellen die Unterschiede beider Dialekte hinsichtlich der Datentypen und Systemfunktionen dar. Es gibt kaum zwei Funktionen in ihnen, die sich exakt gleich verhalten.

Die Tests zur rekursiven Berechnung der Fakultät sind zugleich ein Beispiel für die Verarbeitung ungewöhnlich großer – über eine Bildschirmseite reichender – Zahlen. Auffallend ist, daß bei diesen Zahlen der Unterschied zwischen Compiler und Interpreter nahezu bedeutungslos wird. Bei den anderen Funktionen ist er jedoch gravierend – die kompilierten Funktionen sind bis über fünfzigmal schneller.

Bei der Funktion Q, entnommen aus D. R. Hofstadters „Gödel, Escher, Bach" (1985, 149), handelt es sich um eine hochgradig rekursive arithmetische Funktion. Sie wurde in diesen Test nicht zuletzt deswegen aufgenommen, weil die in ihr vorkommenden Zahlen auch von **XLISP** bewältigt werden können.

Queen ist ein Programm zur Lösung einer Denksportaufgabe: wie können acht Damen so auf einem Schachbrett positioniert werden, daß sie sich nicht gegenseitig bedrohen. Das Programm wurde der hervorragenden Lisp-Einführung von P. H. Winston und B. K. P. Horn entnommen (LISP, 2. Auflage, Addison-Wesley 1984). Um bessere Vergleichsmöglichkeiten zu haben, wurde die Aufgabe auf kleinere Felder übertragen. Es ist insofern typisch für Lisp, als seine Abarbeitung über viele, teilweise indirekt-rekursive Funktionsaufrufe, sowie listenverarbeitende Funktionen erfolgt.

Um das Verhalten bei einem etwas komplexeren Programm zu testen, wurde ein ATN-Parser zur Syntax-Analyse „natürlicher" Sprache ausgewählt. Hier zeigt sich die Problematik solcher Vergleichstests besonders deutlich. Ursprünglich in **XLISP** geschrieben, mußten wichtige in **XLISP** bereits implementierte Funktionen für die **Cambridge Lisp**-Version von Hand geschrieben werden. Eine die Möglichkeiten von **Cambridge Lisp** besser nutzende Version ist etwas effizienter. Während sie interpretiert aber immer noch langsamer als das **XLISP**-Programm ist, läuft sie kompiliert über zwanzigmal so schnell.

Gesamteindruck

Mit **Cambridge Lisp** bietet METACOMCO ein ebenso mächtiges wie flexibles und komfortables Programmiersystem an, das meines Erachtens zu den besten Software-Produkten gehört, die derzeit für den Atari ST verfügbar sind.

Zu wünschen ist allerdings, daß die angesprochenen Mängel wie die fehlende GEM-Unterstützung bald behoben werden. Auch würde die Aufnahme des einen oder anderen Konzepts moderner Lisp-Dialekte dieses Produkt bereichern und die Lisp-Programmierung auf dem Atari ST noch attraktiver machen. Ich denke dabei an Flavors zur objektorientierten Programmierung, wie sie bereits von **XLISP** unterstützt wird, an das von **Common Lisp** bekannte Konzept der multiple values sowie die Möglichkeit der lokalen Bindung von Funktionen, wie sie **SCHEME** bietet.

Am positiven Gesamteindruck ändert dies nichts. Kein Zweifel, mit **Cambridge Lisp** läßt sich der Atari ST zu einem Instrument ausbauen, das für ernsthafte KI-Projekte gut gerüstet ist und auch einen Vergleich mit teuereren Systemen nicht zu scheuen braucht.

DAS PROGRAMMPAKET FÜR DEN PROFESSIONELLEN EINSATZ

WARENWIRTSCHAFT
BARCODE-LESER
COMPUTER-KASSE

DATEIVERWALTUNG
Firmenstamm · Artikel · Kunden · Lieferanten · Lager

KASSENBETRIEB
Kassendisplay am Monitor · Kassenbon mit Artikel-Nummer, Bezeichnung, Stückzahl, Rabatt, Skonto, MWSt u.v.a. · Lfd. Zwischensumme, Gegeben, Rückgeld, Kassenschublade · Betrieb mit Barcode-Leser oder Handeingabe

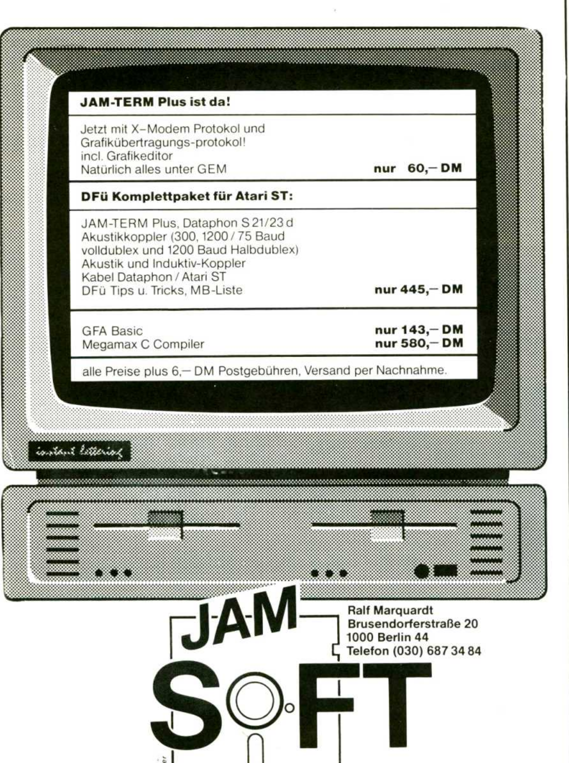
FAKTURIERUNG
Erfassen Aufträge · Druck Auftragsbestätigungen, Lieferscheine, Versandpapiere und Rechnungen

WARENWIRTSCHAFT
Erfassen Anfangsbestände, Wareneingänge, Inventur · Drucken Einkauf, Verkauf, Warenbestand, Bruttoerlös, Fehlbestand, Artikel-Renner, Preislisten, Barcode-Etiketten

Zum sagenhaften Preis incl. Lesestift und Decoder 2 790,– DM
Demo-Diskette mit Handbuch Schutzgebühr (NN) 48,– DM

In Kürze lieferbar:
FIBU mit OP-Verwaltung · DFÜ · Materialwirtschaft · Bestellwesen · Rückstandsverwaltung · u.a.
Fordern Sie unsere Informationsmappe an.

CDS EDV-SERVICE GMBH · Windausstraße 2 · 7800 FREIBURG



ATARI ST

Anwendersoftware	
BS-Lager	269,--
BS-Fakt	589,--
BS-Handel	858,--
VIP-Professional dt. Vers.	449,--
dBMAN Version 2.O2G	399,--
LASERBASE ST	359,--
K-Spread deutsch	179,--
Music Studio	99,--
Sprachen/Entwicklung/Grafik	
GFA Basic Vers. 2.0	149,--
GFA Compiler	149,--
GFA Basic/Compiler zus.	239,--
GFA Vektor 3D Grafik	139,--
GFA Draft CAD Programm	269,--
Metacomco Assembler	159,--
Metacomco Pascal	289,--
Metacomco C-Compiler	319,--
Megamax C-Compiler	449,--
Graphic Artist	649,--
Degas	139,--
Spiele	
Arena	89,--
Borrowed Time	69,--
Brataccas	79,--
Electronic Pool	63,--
Fire Blaster	63,--
Karate Kid II	72,--
Leader Bord Golf	79,--
Little Comp. People	79,--
Silent Service	79,--
Hacker II	79,--
Ultima II	89,--
War Zone	63,--
Winter Games	72,--
World Games	72,--

AMIGA

AMIGA 512 KB kompl. mit Maus u. Farbmonitor	nur 2899,--
Amiga Druckerkabel 2 Meter	39,--
Anwendersoftware	
Deluxe Paint	229,--
Deluxe Print	229,--
Deluxe Video	229,--
Music Studio	119,--
Sprachen/Entwicklung/Grafik	
K-Seka Assembler	249,--
Absoft Fortran Compiler	849,--
Absoft Dynamic CAD	1499,--
Metacomo Lisp	499,--
Metacomco Pascal	299,--
Metacomco Toolkit	99,--
TDI Modula II	329,--
Soft Wood Amiga File	329,--
Spiele	
Arena	89,--
Archon II	79,--
Borrowed Time	79,--
Brataccas	89,--
Deep Space	99,--
Leader Bord Golf	72,--
Little Comp. People	89,--
Hacker II	72,--
Marble Madness	79,--
One on One	79,--
Skyfox	79,--
Temple of Asphai	72,--
Winter Games	72,--
World Games	72,--
SS-50 Diskettenbox mit Schloß für 50 3,5" Disk	nur 24,95

Thomson Farbmonitor CM 36512 incl. Kabel an ATARI ST	nur 849,--
10er Pack 3,5" Disketten 1 DD 135 tpi	ab 39,--
10er Pack 3,5" Disketten 2 DD 135 tpi	ab 49,--

Wir liefern sämtl. Hard- und Software zu äußerst günstigen Preisen!
Sofort kostenlos Preisliste anfordern!

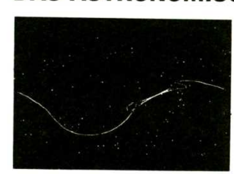
Computer & Zubehör Versand Gerhard und Bernd Waller GbR
Kieler Str. 689, 2000 Hamburg 54, ☎ 040/5706007 + 5705275

DAS SONNENSYSTEM –

spektakulär wie nie zuvor:

SKYPLOT

DAS ASTRONOMISCHE SUPERPROGRAMM FÜR DEN MEGA-ATARI


Die Bahn des Kometen Halley

Informationen über die Planeten unseres Sonnensystems, so z. B. über die Oortsche Wolke, die jenseits des Planeten Pluto im Weltall entdeckt worden ist. Aus diesen fernen Regionen stammen jene Kometen wie der Halley'sche Komet, der derzeit im Anflug in Richtung Erde ist.

Dieses Programm verrät Ihnen alles, was Sie wissen wollen. Nicht nur die Sonne und deren Finsternis, die Milchstraße oder der Saturn werden Sie faszinieren, nein, auch die „Schönheit" des Himmels: der Hantel-Nebel aus dem Sternbild Vulpecula; insgesamt 612 Sterne, 1053 Nebel und Sternhaufen und alle Planeten. Skyplot zeigt Ihnen Ihr Sternzeichen am Himmel, wie es sich zeigt, welche Sterne und Planeten diesem angehören und wo es sich befindet.

Karte umfasst 39.65 x 23.48 Bogensekunden

Mond
Entfernung von der Erde 398250.1 km

Ekliptikale Laenge 305.38, Breite -1.13 Grad
Rektaszension 20.33 h, Deklination -20.54 Grad
Phasenwinkel 151.15 Grad
Mondalter 2.4 Tage
Scheinbarer Aequatordurchmesser 1800.61 Bogensekunden

Die Menüleiste gibt Ihnen einen Überblick über die Vielfalt von Skyplot:

S K Y P L O T

CSM 1986

Autor: Frank Thielen (Vereinigung der Krefelder Sternfreunde)

☐ Himmel darstellen	☐ Einstellung Sterne ja / nein
☐ Eingabe Himmelsdarstellung	☐ Einstellung Planeten ja / nein
☐ Eingabe Beobachtungsdatum	☐ Einstellung Nebel ja / nein
☐ Eingabe Beobachtungszeit	☐ Bewegung eines Himmelskoerpers
☐ Eingabe Beobachtungsort	☐ Konjunktion zweier Koerper suchen
☐ Anzeigen der eingestellten Werte	☐ Sonnensystem darstellen
☐ Sternbilder suchen	☐ Invertierten Himmel zeigen
☐ Planeten, Sterne oder Nebel suchen	☐ Himmelsdarstellung abspeichern
	☐ Sequenz von Bildern abspeichern
☐ Programm beenden	☐ Farben einstellen

Möchten Sie wissen, wann die nächste Mondfinsternis ist?

Möchten Sie wissen, wann der Halley'sche Komet sich der Erde nähert?



Suchen nach Sternbildern

Datum 11. 3. 1986
Zeit 6.00 h
Geogr. Breite -22.30 Grad
Sonnenhoehe -0.17 Grad

Stern Alpha Crucis: Acrux
scheinbare Helligkeit 1.05 Groessenklassen
absolute Helligkeit -3.75 Groessenklassen
Spektralklasse B1n/B3, Doppelstern
Sternbild Kreuz des Suedens Crux Cru
Entfernung 91.0 Parsec (296.7 Lichtjahre)
Rektaszension 12.24 h, Deklination -62.49 Grad
Distanz der Doppelsternpartner 4.7 Bogensekunden
Hoehe ueber dem Horizont 27.41 Grad

☐ 2 Grad ☐ 2 Grad
☐ 10 Grad ☐ 10 Grad
☐ 50 Grad ☐ 50 Grad

Datum 26. 9. 1986
Zeit 2.00 h
Geogr. Breite 50.24 Grad
Sonnenhoehe -33.19 Grad

Jupiter
Entfernung von der Erde 4.006 AE oder 599.3 Mio. km
Entfernung von der Sonne 4.974 AE oder 744.1 Mio. km
Heliozentrische Laenge 348.51, Breite -1.13 Grad
Ekliptikale Laenge 345.32, Breite -1.31 Grad
Rektaszension 23.09 h, Deklination -7.06 Grad
Phasenwinkel 3.21 Grad
scheinbare Helligkeit -2.43 Groessenklassen
Scheinbarer Aequatordurchmesser 49.14 Bogensekunden
Hoehe ueber dem Horizont 19.27 Grad

☐ 2 Grad ☐ 2 Grad
☐ 10 Grad ☐ 10 Grad
☐ 50 Grad ☐ 50 Grad

Möchten Sie wissen, wie sich der Himmel an den verschiedenen Jahreszeiten zeigt?

Möchten Sie mehr wissen über das Ungewisse?

Bestellen Sie jetzt, und Sie können an einer faszinierenden Fülle von Informationen über unser Sonnensystem teilhaben und die Antwort auf viele Fragen erfahren.

BESTELLCOUPON

Einsenden an: **CSM Computer Special Marketing GmbH**
Vorhölzerstraße 18, 8918 Dießen

Hiermit bestelle ich das Programm **Skyplot** auf Diskette zum Micro-Preis von **149,– DM**

Vorname:

Straße:

Name:

PLZ/Ort:

Versandwunsch bitte angeben:
○ per NN
○ per Vorkasse
○ per Vorausscheck

Hardcopy Quer

Dieses Programm bringt Hardcopies im Querformat (wie DEGAS) zu Papier und nutzt die gesamte Papierbreite. Es wird durch 'Alternate+Help' gestartet und ist jederzeit aufrufbar.

Die Fähigkeit der Hardcopy ist eine äußerst wichtige Eigenschaft des ST. Leider ist sie nicht bei allen Druckern anwendbar, da nicht alle Drucker auf dieselben Steuercodes reagieren.

Oft fällt der große Begriff 'EPSON-Kompatibel' auf, der verdeutlichen soll, daß ein solcher Drucker problemlos arbeitet. Allzuoft zeigt sich aber, daß gerade bei der Hardcopy trotzdem irgendetwas nicht geht – meist der Zeilenvorschub, der etwas zu groß oder viel zu klein ist. Sogar bei Epson-Druckern (LQ 800) stimmt er nicht. Hier hilft meist nur ein eigenes Treiberprogramm. 'Hardcopy Quer' ist zwar auf einen Druckertyp zugeschnitten, kann aber durch Änderung der Steuercodes an jeden Drucker angepaßt werden. Diese Steuercodes sind sowohl beim Assemblerprogramm als auch beim zugehörigen Basiclader leicht sichtbar und nach einem Blick in das Druckerhandbuch für die eigenen Bedürfnisse abänderbar. Eine äußerst wichtige Eigenschaft dieses Treibers ist, daß er, wie die normale Hardcopy auch, durch Drücken von ALTERNATE+HELP ausgelöst wird und somit ständig zur Verfügung steht.

Bedienung

Das Programm wird vom Desktop aus gestartet und kehrt sofort zu ihm zurück. Von jetzt an wird die neue Hardcopyroutine benutzt, die folgendermaßen bedient wird:

ALTERNATE/HELP, beliebige Taste außer "2" und "3":
Hardcopy im Querformat (576 Punkte Grafik)

ALTERNATE/HELP , "2":
Hardcopy im Querformat mit doppeltem (vertikal leicht verschobenem) Anschlag

ALTERNATE/HELP , "3":
Hardcopy im Querformat mit vierfachem (vertikal und horizontal leicht verschobenem) Anschlag

Der Druckvorgang kann durch nochmaliges Drücken von ALTERNATE/HELP abgebrochen werden. Außerdem wird das Programm nach 30 Sekunden abgebrochen, falls kein Drucker im "ON LINE"-Betrieb angeschlossen ist. Eventuelle Änderungen der Druckercodes können im Basic- und Assembler-Listing leicht vorgenommen werden.

Assembler-Listing "HARDCOPY.S":

Der Quelltext kann direkt mit dem Entwicklungssystem Assembler von DIGITAL RESEARCH assembliert werden. Das Umschreiben auf andere Assembler dürfte keine Probleme bereiten. Ich will hier näher auf das Listing eingehen:

Zuerst gehe ich mit der GEMDOS-Funktion 32 in den Supervisor Modus und rette anschließend den alten Supervisor Stackpointer nach d6. Danach hole ich mir mit der XBIOS Funktion 2 die physikalische Bildschirmadresse und bringe sie in a0. Wenn ich nun $7d00

```
*********************
* HARDCOPY          *
* Kai Toedter 1986 *
*********************

begin:          clr.l    -(sp)
                move     #32,-(sp)          * supervisor mode
                trap     #1
                addq.l   #6,sp
                move.l   d0,d6              * alter stack
                move     #2,-(sp)           * phys_base
                trap     #14
                addq.l   #2,sp

                movea.l  d0,a0              * Programm hinters Video Ram
                adda     #$7d00,a0          * kopieren
                lea      (a0),a2
                lea      start,a1
                move.l   #ende-start-1,d0
loop:           move.b   (a1)+,(a0)+
                dbra     d0,loop

                movea.l  $456,a0            * _vblqueue
                adda     #28,a0
                move.l   a2,(a0)            * neuer Vector
                move.l   d6,-(sp)           * alter stack
                move     #32,-(sp)          * user mode
                trap     #1
                addq.l   #6,sp
                clr.l    -(sp)
                trap     #1
start:          tst      $4ee               * pflag
                beq      copy
                rts

copy:           move     #2,-(sp)           * phys_base
                trap     #14
                addq.l   #2,sp
                move.l   d0,screen

char_in:        move     #7,-(sp)           * direct con without echo
                trap     #1
                addq.l   #2,sp
```

addiere, zeigt a0 genau hinter das Video Ram. Diesen Wert rette ich nach a2. Danach kopiere ich das eigentliche Hardcopy Programm (das bei "start" beginnt und bei "ende" aufhört) hinter das Video-Ram. Dort befinden sich 768 unbenutzte Bytes, die auch erhalten bleiben. Steht unser Programm dort, können wir zu jeder Zeit und von jedem Programm aus (das natürlich nicht die 768 Bytes benutzt oder unseren Interrupt sperrt Hardcopys machen. Weiter zum Programm: Speicherstelle $456 zeigt auf eine Liste von Vektoren, die nach einem VBL-Interrupt (Bildrücklauf) angesprungen werden. Ich lade ihren Inhalt nach a0 (das geht nur im Supervisor Modus), zähle 28 dazu (also zeigt a0 jetzt auf den achten Vektor, der auf die alte Hardcopy-Routine zeigt) und ersetze den alten Vektor durch meinen neuen. Nun packe ich den geretteten Stackpointer auf den Stack und gehe wieder in den User Modus. Bei "start" beginnt das eigentliche Hardcopyprogramm. Zuerst teste ich die Speicherstelle $4ee, die bei gleichzeitigem Drücken von ALTERNATE und HELP um eins erhöht wird. Bei einem Wert gleich Null soll eine Hardcopy gemacht werden, sonst mit RTS zum Aufruferprogramm zurückgesprungen werden. Weiter geht's mit "copy". Mit der XBIOS Funktion 14 hole ich mir die physikalische Bildschirmadresse und rette sie nach "screen". Die GEMDOS Funktion 7 wartet auf einen Tastendruck und liefert den ASCII-Wert im untersten Byte von d0. Jetzt werden noch "doppelx" und "doppely" auf Null gesetzt. Nun wird d0 auf "2" oder "3" getestet und die entsprechenden Werte in "doppelx" und "doppely" gebracht. Bei "_in" geht's nun richtig los. d6 ist der Spaltenzähler. Da ein Bildschirm aus 80 Spalten á 400 Bytes besteht, wird d6 mit Spalten − 1 = 79 geladen. An dieser Stelle möchte ich kurz die kleinen Unterprogramme erklären, da eine weitere Programmbeschreibung vielleicht verwirrend sein könnte.

out:

Ein Zeichen in d3 wird mit der GEMDOS Funktion 5 an den Drucker ausgegeben. Wenn alles glatt lief (d0 = −1), rts, sonst Rücksprung zum Aufruferprogramm. (Da sich "out" in der zweiten Unterprogrammverschachtelung befindet, einfach 2★4=8 zum Stack addieren). Das geschieht z. B., wenn kein Drucker im "ON LINE"-Modus ange-

```
            move     #0,doppelx
            move     #0,doppely

            cmpi.b   #"2",d0
            bne      tst1
            move     #1,doppelx
            bra      _in
tst1:       cmpi.b   #"3",d0
            bne      _in
            move     #1,doppely
            move     #1,doppelx
_in:        move.l   #79,d6
            movea.l  screen,a5
            suba.l   #81,a5
lo:         jsr      spalte

            cmpi     #0,doppely
            beq      lo1

            sub.l    #1,a5
            jsr      pix_vor
            jsr      spalte
lo1:        cmpi     #0,doppelx
            beq      lo3
            jsr      klein
            sub.l    #1,a5
            jsr      spalte
            cmpi     #0,doppely
            beq      lo3
            sub.l    #1,a5
            jsr      pix_vor
            jsr      spalte
lo3:        jsr      gross
            tst      $4ee
            bne      exit
            dbra     d6,lo


exit:       move     #-1,$4ee           * Hardcopy fertig
            move.l   #4,d1
            lea      _exit(pc),a2
            bra      gr_in1


spalte:     adda.l   #32001,a5          * eine der 80 Spalten drucken
            move.l   #399,d4            * 400 horizontale Bytes
            jsr      gra_in
spa_1:      clr      d3
            move.b   (a5),d3
            suba.l   #80,a5
            jsr      out
            dbra     d4,spa_1
            rts


out:        move     d3,-(sp)           * Zeichen in d3 an Drucker ausgeben
            move     #5,-(sp)
            trap     #1
            addq.l   #4,sp
            tst      d0
            bne      out_ok
            addq.l   #8,sp
out_ok:     rts


gra_in:     lea      graf_on(pc),a2
gr_in1:     move.b   (a2)+,d3           * Den jeweiligen Code an Drucker
            cmpi.b   #255,d3            * ausgeben
            beq      gr_ex
            jsr      out
            bra      gr_in1
gr_ex:      rts

pix_vor:    lea      gpix_vor(pc),a2
            bra      gr_in1

klein:      lea      kl_linef(pc),a2
            bra      gr_in1

gross:      lea      gr_linef(pc),a2
            bra      gr_in1
```

schlossen ist, da die Funktion dann nach 30 Sekunden den Wert 0 in d0 liefert.

gr__in1:
Der Code, auf den a2 zeigt, wird byteweise ausgegeben, bis (a2)=255 ($FF) ist.

gra__in,pix__vor,klein,gross:
Übergeben graf__on,gpix__vor,kl__linef,gr__linef (die den Blöcken 1–4 im Basic-Listing entsprechen) an gr__in1.

spalte:
Eine Spalte (= 400 Bytes) wird von "unten" nach "oben" ausgegeben. d4 ist der Bytezähler. Da sich a5 am Ende "ganz oben" in einer Spalte befindet, muß 32 001 addiert werden, um auf das "unterste" Byte der nächsten Spalte zu kommen. Da ja eine Bildschirmzeile bekanntlich aus 80 Bytes besteht, wird von a5 bei jedem Durchlauf 80 abgezogen.

exit:
In die Speicherstelle $4ee wird –1 gepackt, damit beim nächsten Drücken von ALT/HELP wieder eine Hardcopy ausgelöst wird. Danach wird noch "exit" (entspricht Block 5 im Basic-Listing) an gr__in1 übergeben und mit dem letzten RTS ins Aufruferprogramm gesprungen.

So, jetzt geht's endlich weiter im Hauptprogramm und zwar bei "10". Nachdem die erste Spalte gedruckt wurde, wird "doppely" auf Null geprüft. Bei ungleich Null wird einfach eine Spalte zurückgegangen, indem 1 von a5 abgezogen wird, und die Ausgangsspalte leicht verschoben (wegen "pix__vor") noch einmal gedruckt. Bei "101" wiederholt sich das ganze mit "doppelx". (Mit "klein" wird ein 1/216 Inch Zeilenvorschub ausgeführt). Weiter geht's mit "103". Da jetzt die Spalte auf jeden Fall fertig ist, wird mit "gross" ein 24/216 Inch-Zeilenvorschub ausgeführt. Nun wird noch getestet, ob ALT/HELP zum zweiten Mal gedrückt wurde. In diesem Fall wird die Hardcopy vorzeitig beendet ("exit"), andernfalls wird die Schleife solange durchlaufen, bis die Hardcopy ausgegeben wurde.

```
**************************************************************************
* Druckercodes und Daten. Die Druckercodes werden mit "255" auf 9 Byte Laenge *
* aufgefuellt, damit sie im Basic-Listing leichter geaendert werden koennen.  *
**************************************************************************

doppelx:        .dc.w   0                                 * X-Verdopplung
doppely:        .dc.w   0                                 * Y-Verdopplung
screen:         .dc.l   1                                 * phys. Bildschirmadr.

graf_on:        .dc.b   13,27,42,5,144,1,255,255,255      * CR+ 576 Grafik
gpix_vor:       .dc.b   13,27,42,1,1,0,0,255,255          * CR+ 960 Grafik
kl_linef:       .dc.b   27,51,1,10,255,255,255,255,255    * LINE FEED 1/216  Inch
gr_linef:       .dc.b   27,51,24,10,255,255,255,255,255   * LINE FEED 24/216 Inch
_exit:          .dc.b   13,10,27,65,12,255,255,255        * CR+LF+LF=1/6 Inch

ende:           .dc.b   255
                .end
```

Der Basic Lader "HARDCOPY.BAS":

Wird das Programm nach dem Eintippen mit "RUN" gestartet und es erscheint "Alles klar" im OUTPUT-Fenster, so wurde das Programm "HARDCOPY" auf Diskette erzeugt, das ganz normal mit Doppelklick gestartet werden kann. Erscheint "Data Fehler !", so befindet sich ein Tippfehler in einer der Data-Zeilen.

Nach dem Programmcode kommen 5 Blöcke Druckercodes. Jeder dieser Blöcke muß immer aus genau neun Bytes bestehen. Wenn der jeweilige Druckercode aus weniger als neun Bytes besteht, muß der Block mit "FF"s auf neun Byte aufgefüllt werden. Dabei ist zu beachten, daß das neunte Byte immer ein "FF" sein muß. Der eigentliche Druckercode kann also aus maximal acht Bytes bestehen.

Block 1
besteht aus einem CR (CARRIAGE RETURN = Wagenrücklauf) und einer Sequenz, die den Drucker darauf

```
10    '    *********************
20    '    * HARDCOPY 3        *
30    '    * Kai Toedter 1986 *
40    '    *********************
50    '
60    open "R",1,"HARDCOPY.PRG",16
70    field#1,16 as b$
80    a$=""
90    for n=1 to 16: read d$: if d$="x" then 590
100   z=val("&H"+d$): a$=a$+chr$(z): sum=sum+z: next n
110   lset b$=a$: c=c+1: put 1,c: goto 80
120   '
130   data 60,1A,00,00,01,90,00,00,00,00,00,00,00,00,00,00
140   data 00,00,00,00,00,00,00,00,00,00,00,00,42,A7,3F,3C
150   data 00,20,4E,41,5C,8F,2C,00,3F,3C,00,02,4E,4E,54,8F
160   data 20,40,D0,FC,7D,00,45,D0,43,F9,00,00,00,48,20,3C
170   data 00,00,01,45,10,D9,51,C8,FF,FC,20,79,00,00,04,56
180   data D0,FC,00,1C,20,8A,2F,06,3F,3C,00,20,4E,41,5C,8F
190   data 42,A7,4E,41,4A,79,00,00,04,EE,67,02,4E,75,3F,3C
200   data 00,02,4E,4E,54,8F,23,C0,00,00,01,5E,3F,3C,00,07
210   data 4E,41,54,8F,33,FC,00,00,00,00,01,5A,33,FC,00,00
220   data 00,00,01,5C,0C,00,00,32,66,0A,33,FC,00,01,00,00
230   data 01,5A,60,16,0C,00,00,33,66,10,33,FC,00,01,00,00
240   data 01,5C,33,FC,00,01,00,00,01,5A,7C,4F,2A,79,00,00
250   data 01,5E,9B,FC,00,00,00,51,61,56,0C,79,00,00,00,00
260   data 01,5C,67,08,53,8D,61,00,00,8C,61,44,0C,79,00,00
270   data 00,00,01,5A,67,1A,61,00,00,82,53,8D,61,32,0C,79
280   data 00,00,00,00,01,5C,67,08,53,8D,61,00,00,68,61,20
290   data 61,00,00,6E,4A,79,00,00,04,EE,66,04,51,CE,FF,BA
300   data 33,FC,FF,FF,00,00,04,EE,72,04,45,FA,00,86,60,36
```

vorbereitet, 400 Bytes in 576 Punkte Grafik auszugeben, bei der bei den meisten Druckern ein horizontales/vertikales Verhältnis von 1/1 besteht. Wenn die Hardcopy vertikal gestaucht werden soll, kann die Sequenz z. B. in 640 Punkte Grafik umgeändert werden.

Block 2
wird nur bei vertikaler Verdoppelung (Modus 2 und 3) benutzt. In diesem Fall wird ein "0" Byte in 960 Punkte Grafik gedruckt, so daß eine leichte vertikale Verschiebung bei der fertigen Hardcopy auftritt.

Block3
wird nur bei horizontaler Verdoppelung (Modus 3) benutzt. Er besteht aus der Sequenz für die Umstellung des Zeilenvorschubs (LF = LINE FEED) auf 1/216 Inch.

Block 4
besteht aus einem CR und der Umstellung des Zeilenvorschubs auf 24/216 Inch. Dies ist im Programm der Abstand zwischen zwei Druckzeilen. Beim SEIKOSHA SP-800 stimmt dieser Wert, bei anderen Druckertypen muß er eventuell geändert werden.

```
310   data DB,FC,00,00,7D,01,28,3C,00,00,01,8F,61,24,42,43
320   data 16,15,9B,FC,00,00,00,50,61,06,51,CC,FF,F2,4E,75
330   data 3F,03,3F,3C,00,05,4E,41,58,8F,4A,40,66,02,50,8F
340   data 4E,75,45,FA,00,2A,16,1A,0C,03,00,FF,67,04,61,E0
350   data 60,F4,4E,75,45,FA,00,21,60,EC,45,FA,00,24,60,E6
360   data 45,FA,00,27,60,E0,00,00,00,00,00,00,00,01
370   '
380   ' Die Druckercodes muessen mit "FF" auf eine Laenge
390   ' von 9 Bytes aufgefuellt werden !
400   ' ( CR= CARRIAGE RETURN, LF= LINE FEED )
410   '
420   ' Block 1 : 400 Bytes in 576 Punkte Grafik
430   data 0D,1B,2A,05,90,01,FF,FF,FF
440   '
450   ' Block 2 : 1 "0" Byte in 960 Punkte Grafik
460   data 0D,1B,2A,01,01,00,00,FF,FF
470   '
480   ' Block 3 : LF = 1/216 Inch
490   data 1B,33,01,0A,FF,FF,FF,FF,FF
500   '
510   ' Block 4 : LF = 24/216 Inch
520   data 1B,33,18,0A,FF,FF,FF,FF,FF
530   '
540   ' Block 5 : CR,LF,LF = 1/6 Inch
550   data 0D,0A,1B,41,0C,FF,FF,FF,FF
560   '
570   data 00,00,00,00,1E,3E,10,08,0E,10,08,08,10,12,12,00
580   data 00,00,00,00,00,x
590   close 1
600   if sum <> 31019 then print "Data Fehler !": end
610   print "Alles klar !"
```

Block 5
wird bei Beendigung oder beim Abbruch der Hardcopy benutzt. Er besteht aus einem CR, einem LF sowie der Umstellung auf den üblichen Zeilenvorschub von 1/6 Inch.

Kai Tödter

Assemblerkurs

Teil 2

In diesem Teil des Assemblerkurses wollen wir die Adressierungsarten abschließen. Daran anschließend werden wir uns dem Befehlssatz des 68000 zuwenden.

3.) Absolute Adressierung

Mit dieser Adressierungsart hat man die Möglichkeit, direkt auf den Speicher zuzugreifen. Dazu existieren zwei Versionen:

- Absolut kurz
- Absolut lang

Diese beiden Adressierungsarten unterscheiden sich nur in ihrer Länge und Geschwindigkeit. Außerdem ist bei der Adressierungsart die Größe des Adressbereiches auf 64 K-Byte beschränkt.

3.1) Absolut kurz

Die Schreibweise ist $XXXX
oder $FFYYYY,

wobei XXXX oder FFYYYY im Bereich von −32768 und +32767 liegt. Dies bedeutet, daß sich nur zwei Teile des Speichers ansprechen lassen. Der eine Teil liegt im unteren Teil des Speichers (0000−7FFF) und der andere im oberen (FF8000−FFFFFF). Jeder Teil ist jeweils 32 K-Byte groß. Der Adressraum zwischen diesen beiden Teilen kann nicht angesprochen werden.

Beispiel:

NEG.B $1000

	vorher	nachher
1000	12	EE

MOVE.W $1000,$2000

	vorher	nachher
1000	5678	5678
2000	XXXX	5678

In dem ersten Beispiel wird der Inhalt der Speicherstelle $1000 negiert (Zweierkomplement). Wenn bei dieser Adressierungsart Byteverarbeitung vorliegt, so können auch ungerade Adressen angesprochen werden. Im zweiten Beispiel werden die Speicherstellen $1000 und $1001 nach $2000 und $2001 umgespeichert. Ein Zugriff auf eine ungerade Adresse mittels Wortverarbeitung, wie z. B. MOVE.W $1001,$2001, ist hier nicht möglich.

3.2) Absolut lang

Die Schreibweise ist $XXXXXXXX

Diese Adressierungsart erlaubt einem den zugriff auf den gesamten Speicher. Allerdings ist die Schreibweise mit 32 Bit nicht erforderlich, da der 68000 nur 24 Adressleitungen besitzt. Diese Schreibweise würde einen Adressraum von 2 hoch 32, gleich 4 294 967 295, gleich 4 Gigabyte erlauben. Der wahre Adressraum beträgt aber nur 16 Megabyte. Ein weiterer Unterschied zu der kurzen Adressierung ist die Geschwindigkeit. Da der Prozessor einmal mehr auf den Speicher beim Lesen der Adresse zugreifen muß, ist diese Adressierungsart langsamer. Auch hier gilt: Zugriff auf ungerade Adressen nur mittels Byteverarbeitung.

4.) Adressregister indirekt (ARI)

Um die Effizienz in einigen Softwarebereichen zu erhöhen, wurde diese Adressierungsart mit verschiedenen Abwandlungen realisiert. Bei diesen Adressierungsarten enthält ein Adressregister die Adresse der Daten.

Die Gruppe dieser Adressierungsart:

- Adressregister indirekt (ARI)
- ARI mit Postinkrement
- ARI mit Predekrement
- ARI mit Adressdistanz (Displacement)
- ARI mit Index und Adressdistanz

4.1) Adressregister indirekt

Die Schreibweise ist (An)

Mit der geklammerten Schreibweise ist immer der Inhalt der adressierten Adresse gemeint. Damit hat man den Zugriff auf den ganzen Speicher mit den Adressregistern.

Beispiel:

MOVE.W (A1),D3

	vorher	nachher
A1	00001000	00001000
D3	XXXXXXXX	XXXX1234
1000	1234	1234

Bringt den Inhalt der Adresse, die im Adressregister A1 steht, in das Datenregister D3. Das bedeutet, weil in A1 $1000 steht, wird der Inhalt der Adresse $1000 nach D3 gebracht. Der Inhalt von A1, $1000, und dem höherwertigen Wort von D3 bleiben unverändert.

Die Anwendung dieser Adressierungsart liegt z. B. in der Abarbeitung von Sprungtabellen mittels eines JMP (An).

4.2) ARI mit Postinkrement

Die Schreibweise ist (An)+

Die Wirkung dieser Adressierungsart ist im Prinzip genauso wie bei der Adressregister indirekt. Nachdem die Adresse über das Adressregister An angesprochen wurde, wird diese erhöht. Um wieviel es erhöht wird, hängt von der Verarbeitung ab. Ist es Byteverarbeitung, so wird es um 1 erhöht. Wortverarbeitung ergibt eine Erhöhung um 2 und Langwortverarbeitung eine um 4.

Beispiel:
MOVE.W #$1234,(A3)+

	vorher	nachher
A3	000010000	00001002
1000	XXXX	1234

Dieses Beispiel zeigt das Abspeichern einer Konstanten nach einer Adresse, die im Adressregister A3 angegeben ist. Anschließend wird dieses um 2 erhöht, da Wortverarbeitung vorliegt.

Eine Anwendung findet diese Adressierungsart zur Nachbildung von Stackpointern, zum Abarbeiten von Tabellen und um Daten zu verschieben (Blockmoves).

. . . DIGITALISIEREN LEICHT GEMACHT MIT

Maus System

ges. gesch.

KOMPATIBLER GEHT ES NICHT!

DM 98,–

Maus System
- an die Maus anbauen, egal ob ST, Amiga oder Macintosh und schon braucht man nur noch abzuzeichnen
- Vorlagengröße A6 – A0
- kompatibel zu jeder Grafik- und Malsoftware
- zukunftsicher, mit Zubehör ausbaufähig

Maus System DM **98,–**
Maus System Linsensatz DM **59,–**

MonoSTar
das Super-Malprogramm mit Objektmodulation
s. Test Data Welt 11/86 DM **99,–**

ColorSTar
gleiches Programm für Farbmonitor mit unvorstellbaren Möglichkeiten
z. B. – Trommelspiegelung
– das Mischen zweier benachbarter Farben möglich (16450 Farben)
DM **99.–**

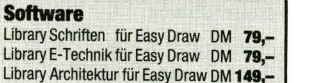
Hardware
Netzwerk
- Netzsystem zum Vernetzen von bis zu 128 Atari-Stationen über DMA (1 Megabit / sec.)
- Passwortschutz
- Mailboxsystem
- Cachespeicher

Masterplatz: DM **1098,–**
pro Slave: DM **178,–**

Festplatten
Qualitätsprodukt aus deutscher Fertigung
20 MB form. DM **1950,–**
40 MB form. DM **3450,–**
85 MB form. DM **6900,–**

Sound Sampler
- 10 bit Auflösung
- incl. Gem-Software DM **498,–**

Video Digitizer
- liest auch Fernsehbilder ein
- 640 × 400 Pixel DM **498,–**



Software
Library Schriften für Easy Draw DM **79,–**
Library E-Technik für Easy Draw DM **79,–**
Library Architektur für Easy Draw DM **149,–**
Easy Draw incl. Schriften- oder E-Technik Libr. DM **398,–**
GFA Basic incl. Konvertierer ST auf GFA Basic DM **149,–**
10 Markendisketten 3.5 DD doppelseitig form., geprüft, incl. 10er Flip-Box DM **69,–**
dto. nur mit 5 Disketten DM **39,–**
CAD Software a. Anfrage

Branchenlösungen für
- Ärzte – Zahnärzte
- Rechtsanwälte – Groß- u. Einzelhandel
- Grafik + CAD – andere a. Anfrage

incl. Support und Schulung

Distributor für die Schweiz:
Münch-Verlag AG
Bungertstr. 15
CH-8802 Kilschberg/Zürich
Tel. 0041/17155702

digital project

P.O.BOX 106169 – D-2800 Bremen 1 – Tel. 0421 – 591220
Händleranfrage erwünscht / Distributoren fürs Ausland gesucht

BESTELL-COUPON
○ Katalog DM 2,– (Schutzgeb. in Briefm.) ○ Ich bestell per NN (+ DM 7,–)
Name
Straße
Ort

4.3) ARI mit Predekrement

Die Schreibweise ist –(An)

Wenn man die letzten zwei Beispiele verstanden hat, so werden Sie anhand dieser Schreibweise schon die Funktion erkennen.

Beispiel:
MOVE.L D7,–(A5)

	vorher	nachher
A5	00001004	00001000
D7	12345678	12345678
1000	XXXX	1234
1002	XXXX	5678

Um Ihnen hier noch einmal die Darstellung des Speichers zu erläutern, habe ich ihn in gerade Adressen aufgeteilt. Jede dieser Adressen ist gemäßt der Datenbusbreite 16 Bit lang. Die höherwertigen 8 Bit sind in den geraden Adressen und die Niederwertigen in den Ungeraden.

Als erstes wird bei einem ARI mit Predekrement das entsprechende Adressregister, um die Zahl der Bytes die verarbeitet werden sollen, erniedrigt. An dieser Adresse kann dann das Datenregister abgespeichert werden.

Auch hier gelten die gleichen Anwendungsgebiete wie unter 4.2 beschrieben. Hier an dieser Stelle möchte ich noch eine wichtige Anmerkung machen. Wird bei Byteverarbeitung das Register A7 bzw. SP angesprochen, so wird um 2 in- bzw. dekrementiert, um den SP auf geraden Adressen zu halten!

4.4) ARI mit Adressdistanz

Die Schreibweise ist d16(An)

Die Angabe d16 steht für ein Displacement (Adressdistanz) von 16 Bit. Diese 16 Bit Zahl ist vorzeichenbehaftet (Zweierkomplement). Mit dieser Adressierungsart hat man die Möglichkeit, den Adresszeiger um einen konstanten Wert zu versetzen.

Beispiel:
MOVE.W $20(A2),$6000

A2	000040000	00004000
4020	8765	8765
6000	XXXX	8765

Adressrechnung:

Adressregister	An	$00004000
Displacement	d16	$00000020
EA		$00004020

Mit dieser Adressierungsart wird zum erstenmal eine etwas aufwendigere Adressrechnung notwendig. Wie Sie sehen, wird der Inhalt des Adressregisters, mit dem Displacement einfach addiert. Dies ergibt die Adresse, deren Inhalt nach $6000 gespeichert wird.

Die Leistungsfähigkeit einer solchen Adressierung zeigt sich in der Vielfalt der Anwendungsmöglichkeiten. Man ist damit in der Lage, z. B. einen Peripheriebaustein mit Werten aus einer Tabelle zu versorgen, wobei die Adressdistanz die verschiedensten Anwendungen ermöglicht. Oder man will, mittels eines Zeigers, auf zwei oder mehrere Tabellen zugreifen, usw...

4.5) ARI mit Index und Adressdistanz

Die Schreibweise ist d8(An,Rx.X)

Diesmal darf das Displacement nur 8 Bit lang sein. Auch diese Zahl ist vorzeichenbehaftet. Außerdem kommt noch ein beliebiges Register (Index) mit einer Angabe der Verarbeitungslänge hinzu. Der Index ist ebenfalls vorzeichenbehaftet.

Beispiel:
MOVE.L $F0(A6,D3.W),$3000

	vorher	nachher
A6	00004000	00004000
D3	15423008	15423008
6FF8	ABCD4321	ABCD4321
3000	XXXXXXXX	ABCD4321

Adressrechnung:

Adressregister	An	$00004000
Index	Rx.W	$00003008
Displacement	d8	$FFFFFFF0
EA		$00006FF8

Bei dieser Adressierungsart ist die Adressrechnung noch etwas aufwendiger. Hier sehen Sie auch den Umgang mit vorzeichenbehafteten Zahlen. Rechnet man mit ihnen, so muß man sie in ihrer Stellenzahl auf die Bitbreite (32 Bit) des Ergebnisses erweitern. Es gelten die Regeln zur Addition mit Zweierkomplementzahlen.

Oft wird man diese Adressierungsart in der Tabellenbearbeitung einsetzen, mit der man variabel auf mehrere Tabellen zugreifen möchte. Dabei zeigt das Adressregister auf den Tabellenanfang, das Indexregister auf den Wert und das Displacement auf die entsprechende Tabelle. Das Displacement kann, wenn es nicht benötigt wird, in der Assemblerschreibweise weggelassen werden. Der Assembler setzt dann als Displacement Null ein.

5.) Programmzählerrelative Adressierung (PCR)

Dies ist eine Adressierungsart, mit der der Atari ST-Anwender normalerweise nicht in Berührung kommt. Das Anwendungsgebiet dieser Adressierungsart ist ein System, daß ohne eine MMU arbeitet und das programmunabhängig von der Position im Speicher laufen soll. Diese Art der Programmierung nennt man PIC (Position-Independent-Code). Gute Assembler bieten die Möglichkeit, normale absolute Programme als programmzählerrelative Programme zu übersetzen. Der Anwender braucht sich dadurch keine Gedanken über die Lauffähigkeit zu machen.

Systeme oder Maschinen, die nicht die Voraussetzung lieferten, die Programme oder Programmodule einfach und schnell zu verwalten, benötigten vom Betriebssystem aus einen PIC-Programm. Diese Programme machten einem die Programmierung schwerer, aber die Verwaltung war schneller, da die Programme im Speicher nicht verschoben werden mußten.

Wie schreibt man PIC?

Da man den Adressbereich des Programmes nicht kennt, während man es schreibt, muß man auf die Verwendung von absoluten Adressen, die sich auf das Programm beziehen, verzichten. Sämtliche Sprungbefehle und Adressrechnungen müssen daher immer auf den Programmzähler bezogen sein.

5.1) PCR mit Adressdistanz

Die Schreibweise ist d16(PC)

Den Trick, der bei dieser Adressierungsart angewendet wird, will ich Ihnen an einem kleinen Beispiel erläutern. Nehmen wir einmal an, wir stünden in einer Straße vor der Hausnummer 10. Drei Häuser weiter, in der Hausnummer 16, wohnt ein Freund von uns. Will nun ein anderer Freund wissen, wo mein Freund wohnt, sage ich ihm „Hausnummer 16". Mittlerweile aber haben die Leute von der Stadtverwaltung die Hausnummern ändern lassen. Daraus folgt, daß meine Freunde sich nicht treffen werden. Hätte ich aber gesagt: „Er wohnt drei Häuser weiter als ich", so hätten sie sich gefunden, unabhängig von der Hausnummer.

Beispiel:

MOVE.W $0A(PC),D6

	vorher	nachher
PC	00003000	00003004
D6	XXXXXXXX	XXXXFEDC
300C	FEDC	FEDC

Adressrechnung:

Programmcounter	PC	$00003000
Plus 2		2
Displacement	d16	$0000000A
EA		$0000300C

Das Displacement ist vorzeichenbehaftet. Dadurch ist es möglich, Daten vor und nach dem Befehl anzusprechen. Dieser Offset errechnet sich aus der Differenz zwischen dem PC+2 und der Adresse. Egal, in welchen Speicherbereich man das Programm verschiebt, die Differenz bleibt immer dieselbe.

5.2) PCR mit Index und Adressdistanz

Die Schreibweise ist d8(PC,Rx.X)

Ebenso wie in 4.5 und den Informationen aus 5 über die PCR-Programmierung, ist diese Adressierungsart zu erklären.

Beispiel:

MOVE.W $FE(PC,A5),D3

	vorher	nachher
PC	00005000	00003004
A5	006A0000	006A0000
D3	XXXXXXXX	00112233
6A5000	00112233	00112233

Adressrechnung:

Programmcounter	PC	$00005000
Plus 2		2
Index	Rx.L	$006A0000
Displacement	d8	$FFFFFFFE
EA		$006A5000

Auch hier findet eine Adressrechnung durch Addition mit dem Programmcounter+2, und dem Index und Displacement statt, die ja bei der Addition vorzeichenrichtig auf 32 Bit erweitert worden sind.

Nun haben wir alle Adressierungsarten besprochen und können uns jetzt den Befehlen zuwenden. Dies ist wieder ein recht großer Block.

Der Befehlssatz des 68000

Um die insgesamt 56 Befehle des 68000 zu beschreiben und zu erläutern, werden wir die Befehle erst einmal in bestimmte Kategorien zusammenfassen. Die Einteilung der Gruppen erfolgt aufgrund ihrer Funktion. Dadurch ergeben sich die folgenden sieben Gruppen:

- Arithmetische Befehle
- BCD Befehle
- Logische Befehle
- Schiebe- und Rotier-Befehle
- Bitmanipulationsbefehle
- Datenübertragungsbefehle
- Programmsteuerbefehle

In der Beschreibung der Befehle werde ich Ihnen die Funktion und Wirkungsweise, die Assemblerschreibweise, die Beeinflussung der Flags, die Größe der Operanden, die zugelasenen Adressierungsarten und die Anwendungsmöglichkeiten aufzeigen.

Einige Befehle will ich Ihnen außerhalb der Reihe schon vorab erklären, da sie für die Programme in diesem Kurs von Bedeutung sind.

Der **Move** Befehl

Der Move Befehl ist wohl der am meisten und am vielseitigsten gebrauchte Befehl überhaupt. Er gehört zu der Gruppe der Datenübertragungsbefehle. Diesen Befehl hatte ich im Teil 1 schon angesprochen, nun folgt aber die ausführliche Beschreibung in allen Varianten. Zu diesen Varianten gehört:

MOVE
MOVEA
MOVEQ
MOVEM
MOVEP
MOVE to CCR
MOVE to SR
MOVE fr SR
MOVE USP

Sie sehen, dieser eine Befehl hat viele verschiedene Aufgaben, die in der Schreibweise deutlich gemacht wird.

Da ich den Befehl in seiner Funktion schon teilweise erklärt habe, werde ich die Beschreibung des Befehls nur noch zusammenfassen. Der MOVE-Befehl übernimmt alle Aufgaben der Datentransporte. Mit ihm lassen sich die Register laden, Daten vom Speicher holen oder dort speichern und Daten zwischen den Registern oder dem Speicher hin- und hertransportieren. Der MOVEA-Befehl wird benutzt, wenn ein Adressregister das Ziel der Operation ist. Der MOVE-Befehl erlaubt fast alle Kombinationen der Adressierung, außer Adressregister direkt als Ziel. Dies ist in der exakten Schreibweise dem MOVEA-Befehl vorbehalten.

Ebenso wie der MOVEA sind der MOVEM und MOVEP spezielle Befehle. Der MOVEM (M=Memory) wird benutzt, um einen, mehrere oder alle Daten- und Adressregister des 68000 im Speicher abzulegen und wieder zu holen. Die Auswahl der Register erfolgt durch die Angabe des jeweiligen Registers, das durch ein / getrennt oder in einem Block durch einen – getrennt wird.

Beispiel:

MOVEM.W A7/A5/D3-D6,-(SP)

A7	00001100
10FE	A7
10FD	A5
10FB	D6
10FA	D5
10F8	D4
10F6	D3

Die Reihenfolge läßt sich nicht bestimmen. Sie ist immer D0-D7 und dann A0-A7. Die einzige Ausnahme ist die Adressierungsart ARI mit Predekrement als Ziel. So wird in unserem Beispiel erst A7, dann A5,D6,D5,D4 und D3 auf den Stack abgelegt. Genau in der anderen Reihenfolge.

Mit dem MOVEP-Befehl (P=Peripherie) kann man Daten zu oder von den alternierenden Adressen des Speichers oder der Peripherie gespeichert oder geholt werden. Der MOVEP-Befehl überträgt die Daten immer byteweise, dadurch ist es möglich, die 8-Bit Peripherie einfach zu versorgen. Die 8-Bit-Peripherie ist so organisiert, daß die Adressen der Peripherieregister im 16 Bit breiten Speicher immer um zwei differieren und somit immer an geraden oder ungeraden Adressen liegen. Der Befehl schreibt oder liest somit immer byteweise an den geraden oder ungeraden Adressen. Diese byteweise Übertragung läuft so, daß das höchstwertige Byte des Datenregisters zuerst übertragen wird, und das niedrigstwertige zuletzt.

Beispiel:

MOVEP.W D3,5(A3)

A3	005000
D3	1234
5004	XX12
5006	XX34

Hier wird das Datenregister D3 an die mit ARI und Displacement adressierten Adresse geschrieben. Dabei sieht man deutlich das Schreiben an ungeraden Adressen, wobei die geraden Adressen unverändert bleiben, ebenso wie das Adressregister A3.

Die nächste Gruppe des MOVE-Befehls dient dazu, das Statusregister (SR) bzw. das Condition-Code-Register (CCR) zu lesen oder zu schreiben. Ebenso benötigt man einen Befehl, um im Supervisormodus an den Userstackpointer (USP) zu gelangen. Der MOVE to SR und der MOVE USP Befehl sind privilegierte Befehle. Dies bedeutet, sie dürfen nur im Supervisor Modus angewendet werden. Verletzt man diese Regel, so führt dies zu einer Ausnahmebehandlung (Exception).

Der MOVE to CCR arbeitet nur mit Wortdaten, kann aber nur die untersten fünf Bit des CCR verändern. Er dient zur gezielten Vorbesetzung der Flags des CCR.

Beispiel:

```
MOVE.W #%11001,CCR
            XNZVC (Flags)
```

Ebenso arbeitet der MOVE to SR Befehl, wobei er aber alle funktionalen Bits des SR verändern kann. Die restlichen, nicht benötigten Bits bleiben Null. In der lesenden Richtung arbeitet der MOVE from SR Befehl. Mit diesem Befehl kann ein Anwender den Status des Systems erfahren und entsprechend darauf reagieren.

Beispiel:

```
MOVE.W SR,D3
```

Es wird das Statusregister exakt nach D3 abgebildet, wobei nur das niederwertige Wort benutzt wird. Die restlichen oberen 16 Bits bleiben erhalten.

Mit dem MOVE USP kann der Anwenderstackpointer im Supervisormodus gerettet bzw. mit dem alten oder einem neuen Wert wieder an das User-Programm übergeben werden. Nun bleibt als letztes nur noch der MOVEQ Befehl (Q=Quick=schnell). Mit ihm lassen sich die Datenregister schnell mit einem Startwert von −128 bis +127 belegen. Der Befehl arbeitet immer mit einer 8-Bit-Zweierkomplementzahl und grundsätzlich wie bei der Langwortverarbeitung. D. h., daß das entsprechende Datenregister vorzeichenrichtig erweitert wird.

Nun folgt die Zusammenstellung der erlaubten Verarbeitungsbreite, der Flags, die beeinflußt werden, der erlaubten Adressierungsarten und der Assemblerschreibweise. Die Zeichen unter den Flags bedeuten:

- – bleibt unverändert
- ★ wird entsprechend gesetzt
- U Undefiniert
- 0 wird auf Null gesetzt

Bei der Schreibweise der erlaubten Adressierungsarten, bedeutet:

ARI	alle Adressregister indirekt
abs	Absolut kurz und lang
PCR	alle Programmcounter relativ
SR	Statusregister
CCR	Condition Code Register
USP	User Stack Pointer

Um alle Adressierungsarten zu ermitteln, kann man jede Quelle mit jedem Ziel verknüpfen.

Die nächsten Befehle, die ich Ihnen nun vorstellen möchte, gehören zu der Gruppe der Programmsteuer-Befehle. Diese Gruppe ist eine sehr wichtige, da mit ihnen der Ablauf des Programms gesteuert werden kann. Zu ihnen gehören die Sprung- oder Verzweigungs-Befehle. Diese teilen sich wieder in bedingte und unbedingte Sprünge auf. Hierzu sind auch die Unterprogrammaufrufe und die Systemaufrufe hinzuzurechnen.

- Bcc
- DBcc
- BRA
- BSR
- JMP
- JSR
- RTE
- RTR
- RTS
- TRAP

JMP und BRA

Diese beiden Befehle dienen zum unbedingten Verzweigen in Programmen (ähnlich dem Basic Befehl GOTO). Der Unterschied liegt in der Art des Sprunges. Der BRA (Branch) Befehl arbeitet mit einer 8 oder 16 Bit Adressdistanz (relativ). Dies bedeutet eine Sprungsweite von −128 bis +127 (8-Bit), oder −32767 bis 32768 (16-Bit) vom augenblicklichen Stand des Programmzählers. Der JMP (Jump) Befehl arbeitet absolut. D. h. es wird direkt an die Adresse, die über die EA adressiert worden ist, gesprungen. Man kann dadurch in dem gesamten Speicherraum herumspringen. Mittels der Adressierung der EA, wird der Befehl vielseitig. Es ist auch möglich, über die programmzählerrelative Programmierung, einen relativen Sprung zu machen.

Syntax	Flags XNZVC	.x	Quelle	Ziel
MOVE.x (ea),(ea)	-★★00	B,W,L	Alle	Dn,ARI,abs
MOVEA.x (ea),An	-----	W,L	Alle	An
MOVEQ #Kons,Dn	-★★00	B	#	Dn
MOVEM.x Rlist,(ea)	-----	W,L	Dn,An	abs,ARI/(An)+
MOVEM.x (ea),Rlist	-----	W,L	abs,PCR,ARI/-(An)	Dn,An
MOVEP.x Dn,d(An)	-----	W,L	Dn	D(An)
MOVEP.x d(An),Dn	-----	W,L	d(An)	Dn
MOVE (ea),CCR	★★★★★	W	Alle/An	CCR
MOVE (ea),SR	★★★★★	W	Alle/An	SR
MOVE SR,(ea)	-----	W	SR	Dn,ARI,abs
MOVE USP,An	-----	L	USP	An
MOVE An,USP	-----	L	An	USP

Beispiel:

```
      CLR.L DO
      BRA Marke
      .
      .
Marke MOVE.W D0,D3
      MOVEA.L Marke,A3
      JMP (A3)
      .
      .
Marke MOVE.W D0,D3
```

Mit dem Branch-Befehl kann man das Programm an einer anderen Stelle fortsetzen. Ebenso wäre dies mit Jump möglich gewesen, solange man keinen PIC schreiben muß. Das zweite Beispiel zeigt einen Sprung über das Adressregister A3. Damit können Sie variabel in verschiedene Programmteile verzweigen, indem Sie das Register A3 mit der Adresse des Programmteils versorgen.

Bcc und DBcc

Das Gegenstück zu den unbedingten Verzweigungen sind die bedingten Verzweigungen. Die Verzweigung ist vom CCR abhängig. Welche Bedingung gültig ist, bestimmt folgende Tabelle. Aus dieser Tabelle brauchen Sie nur die entsprechende Mnemonik für cc einzusetzen.

Das mit dem ★ gekennzeichnete cc, F und T, gilt nur für den DBcc Befehl. Manche Assembler erlauben auch DBRA anstatt DBF. Der Unterschied zwischen BGT und BHI, liegt in der Abfrage der Flags. Die mit einem K am Ende der Zeile markierten Bedingungen bedeuten, daß eine Abfrage für die Zweierkomplementarithmetik vorgenommen wird.

Mnemonik	Bedeutung
CC	Carryflag = 0
CS	Carryflag = 1
EQ	Gleich (Z=1)
F ★	Falsch,Nie erfüllt
GE	Größer oder gleich K
GT	Größer K
HI	Höher
LO	Kleiner oder gleich K
LS	Niedriger oder identisch
LT	Kleiner K
MI	Negativ
NE	Ungleich
PL	Positiv
T ★	Wahr, immer erfüllt
VC	Kein Überlauf K
VS	Überlauf K

Die Wirkungsweise

Kommt das Programm an einen Bcc Befehl, so wird zuerst die Bedingung überprüft. Ist diese wahr, so wird an die Stelle verzweigt, die hinter dem Befehl angegeben ist. Dies ist, wie beim BRA Befehl, eine 8- oder 16-Bit Adressdistanz im Zweierkomplement. Ist die Bedingung nicht erfüllt, so setzt das Programm seinen Verlauf mit dem nächsten Befehl fort.

Der DBcc Befehl prüft ebenfalls als erstes die Bedingung. Ist diese Bedingung erfüllt, so wird nicht wie beim Bcc Befehl verzeigt, sondern beim nächsten Befehl das Programm fortgesetzt. Ist die Bedingung nicht erfüllt, so wird das angegebene Datenregister um eins erniedrigt. Ist der Inhalt dieses Datenregisters gleich −1, so geht das Programm zum nächsten Befehl. Ansonsten wird das Programm an dem genannten Label fortgesetzt. Diese interessante Variante wird benutzt, um eine Schleife aufzubauen, die nur mit einer bestimmten Anzahl von Durchläufen abgearbeitet wird. Außerdem kann mit der Bedingung ein Abbruch aus der Schleife erzwungen werden.

Beispiel:

```
Marke Add.L #$FF,D3
      .
      .
      .
      CMP.L #$8000,D3
      BLT Marke
      .
Marke MOVE.W D3,D1
      .
      .
      .
      CMP.W #0,D1
      DBEQ D3,Marke
      .
```

In dem oberen Beispiel wird die Schleife so lange abgearbeitet, bis das Datenregister D3 noch kleiner als $8000 ist. Das untere Beispiel enthält als Abbruchbedingungen:

1.) D1 = 0
2.) D3 = −1

Ist eine dieser Bedingungen erfüllt, so wird der nächste Befehl abgearbeitet. Bei dieser Konstruktion der Schleife muß man beachten, daß die Schleife einmal mehr als der Inhalt von D3 durchlaufen wird. Soll das nicht der Fall sein, so sollte man vor Eintritt in die Schleife das entsprechende Register um eins erniedrigen.

JSR und BSR

Diese beiden Befehle dienen dazu, Unterprogramme aufzurufen (BASIC= GOSUB). Die Ausführung der Befehle unterscheidet sich von den Befehlen JMP und BRA nur in einer winzigen Kleinigkeit. Bevor der Sprung zu der angegebenen Stelle erfolgt, wird die Adresse des nächsten Befehls auf dem Stack abgelegt und der Stackpointer um vier (Langwort) erniedrigt. Mit dieser Adresse ist dann eine Fortsetzung des Programms an dieser Adresse möglich. Dazu gibt es einen Befehl mit verschiedenen Varianten, der den Rücksprung bewirkt.

RTS, RTE und RTR

RTS (Return from Subroutine) beendet jedes Unterprogramm. Dieser Befehl holt sich vom Stack die abgelegte Adresse, erhöht diesen um vier und lädt den Programmzähler mit dieser Adresse. Dadurch wird das Programm an dieser Stelle vorgesetzt. Unterprogramme können beliebig ineinander geschachtelt werden. Benutzt man den Stack, um Daten abzulegen, so muß man ihn vor dem Rücksprung bereinigen, damit die richtige Adresse vom Stack geholt werden kann.

RTE (Return from Subroutine) beendet jedes Unterprogramm. Dieser Befehl holt sich vom Stack die abgelegte Adresse, erhöht diesen um vier und lädt den Programmzähler mit dieser Adrese. Dadurch wird das Programm an dieser Stelle vorgesetzt. Unterprogramme können beliebig ineinander geschachtelt werden. Benutzt man den Stack, um Daten abzulegen, so muß man ihn vor dem Rücksprung bereinigen, damit die richtige Adresse vom Stack geholt werden kann.

RTE (Return from Exception) ist ein privilegierter Befehl. Er dient zum Rücksprung aus Interrupt-, Trap- und Exceptionbehandlungen. Bevor der Rücksprung erfolgt, wird das Statusregister mit dem Wert geladen, der sich auf dem Stack befindet. Dieser muß also, am Anfang der Exception, als erstes auf den Stack gebracht werden.

Der RTR (Return mit Laden der Flags) Befehl funktioniert genauso wie der RTE Befehl. Allerdings wird nur das CCR behandelt. Da der RTR kein privilegierter Befehl ist, kann er auch als

Ein Bestseller

Das GFA-BASIC-Buch

NEU

Die GFA-BASIC-Programmsammlung

Aus dem Inhalt

- Der Umgang mit dem Editor
- Die Vorteile der strukturierten Programmierung ohne Zeilennummern und Pascal-ähnliche Struktur
- Kommunikation mit der Außenwelt (Peripherie)-Floppy, Tastatur-Prozessor, MIDI, der parallele u. serielle Bus, Joystick und Maus
- Mathematik und hohe Genauigkeit
- Schnelle Graphik schnell programmiert – Windows, Sprites, Alertbox, Pull-down Menüs
- Die Systembefehle – einfaches Einbinden von TOS-Befehlen (XBIOS, BIOS, GEMDOS) zur vollständigen Nutzung des Atari-Betriebssystems
- Der Sound – Beschreibung des Soundchips und dessen Programmierung, sogar mit Interrupt-Steuerung



Über 500 Seiten **DM 49,–**

Wichtige Merkmale des Buches

- Die weit über 200 Befehle des neuen GfA-Basic-Interpreters werden mit ihrer Syntax und Bedeutung anhand von vielen Beispielen ausführlich erklärt.
- Das GfA-Basic-Buch ist in Sachgebiete unterteilt, um eine sinnvolle Anwendung der Befehle im Zusammenhang zu ermöglichen. Dabei werden die vielfältigen Fähigkeiten der ATARI ST-Rechner (Graphik, Geschwindigkeit, Window-Technik und Sound) genutzt.
- Das GfA-Basic-Buch ist leicht verständlich geschrieben, so daß auch der Anfänger ohne Probleme seine eigenen Programme erstellen kann.
- Aber auch für den fortgeschrittenen Anwender und für den, der es einmal werden will, bietet das GfA-Basic-Buch die Grundlage zur richtigen Handhabung des Betriebssystems.
- Im Anhang ist eine komplette alphabetische Befehlsübersicht (Nachschlagewerk) enthalten.

★ **Programmdiskette zum Buch: 39,– DM**

---

131 Programme in GFA-BASIC

Folgende Schwerpunkte sind gesetzt:

- Allgemeine Eigenschaften des GFA-Basic
- Zahlen- und Variablenbehandlung
- Dateiverwaltung und Ordner in GFA-Basic
- Grundlegende Befehle
- Schleifentechnik
- Felder (Tabellenverarbeitung u. Matrizen)
- Unterprogramme
- Menü-Steuerung
- Window-Technik
- Zufallszahlen
- Sequentielle Dateien
- RANDOM-Dateien
- Physikalischer Diskettenzugriff
- Textverarbeitung
- Sortierprogramme
- Sortiersystem: SORTMIX
- Fakturiersystem: FAKTURA
- Grafik auf dem Bildschirm u. dem Drucker
- Grafik-Programm: GRAFIKER
- Geometrie, Statistik u. Funktionsplott
- Bildschirmspiel: FLUSS
- Kontoführungsprogramm: KONTO 520
- und viele andere mehr



320 Seiten **DM 49,–**

Wichtige Merkmale des Buches

Anhand von über 130 Programmbeispielen und Erläuterungen lernen und trainieren Sie das Programmieren in GFA-BASIC. Von einfachen, aber grundlegenden Beispielen bis zur ausgereiften Anwendung findet der ST-Besitzer Beispiel-Programme, die den perfekten Einstieg in die Programmiersprache GFA-BSIC leicht und interessant machen.

Alle Beispiele werden so dargestellt, daß das Verständnis für den Programmaufbau gefördert wird, aber auch die Details der GFA-BASIC-Version deutlich werden. Die Programme sind gut dokumentiert, wobei die hervorragenden Möglichkeiten des GFA-BASIC zur strukturierten Programmierung genutzt werden.

★ **Programmdiskette zur Progr.-Sa.: 39,– DM mit 131 Programmen in GFA-BASIC**

Bestellungen bei Ihrem ATARI-Händler (s. Einkaufsführer) oder direkt beim Verlag mit Bestell-Abschnitt

* unverb. empf. Verkaufspreise

Heim-Verlag
Heidelberger Landstr. 194
6100 Darmstadt-Eberstadt
Telefon 0 61 51 / 5 60 57

Bestell-Abschnitt einsenden an:
HeimVerlag, Heidelberger Landstr. 194, 61 Darmstadt-Eberstadt

Ich bestelle:
- ☐ Das GFA-Basic-Buch .... 49,– DM
- ☐ Programm-Diskette zum Buch .... 39,– DM
- ☐ GFA-BASIC Programmsammlung .... 49,– DM
- ☐ Programm-Diskette zur Progr.-Sammlung 39,– DM

zuzügl. 3,– DM Versandkosten (Scheck oder per Nachnahme)

Name: ______________________

Straße: ______________________

Ort: ______________________

Rücksprung aus normalen Unterprogrammen benutzt werden. Auch hier muß das SR auf den Stack abgelegt werden. Trifft das Programm auf ein RTR, so wird nur das CCR wiederhergestellt.

Beispiel:

```
        BSR Marke
        .
        .
        .
Marke MOVEM.L D1-D3/A5,-(A7)
        .
        .
        MOVEM.L (A7)+,D1-D3/A5
        RTS

        JSR Marke
        .
        .
        .
Marke MOVE SR,-(A7)
        .
        .
        RTR
```

Hier in den ersten Beispielen sieht man den Umgang mit Unterprogrammen. Um die Arbeit etwas zu vereinfachen, kann man bei Unterprogrammen eine „Versorgung" und eine „Entsorgung" definieren. Zum Beispiel benötigt ein Unterprogramm als Versorgung die Adresse eines Puffers in A1, einige Register, mit denen man arbeitet, und als Entsorgung das Register D0, das anzeigt, ob die Operation erfolgreich verlaufen ist. Da die Regisdter zum Arbeiten auch im Hauptprogramm verändert sind, sollte man sie vorher abspeichern, und am Ende des Unterprogramms wieder laden, damit im Hauptprogramm die alten Werte wieder verfügbar sind. Sollen ebenfalls die Flags gerettet werden, so bietet sich die nächste Konstruktion an. Als erstes wird das SR auf den Stack gebracht, um danach freie Hand zu haben.

TRAP

Durch diesen Befehl wird eine Exceptionsbehandlung ausgelöst. Dieser Befehl bewirkt, daß der Inhalt des PC und des SR auf den Stack gerettet werden. Danach wird der PC mit der Adresse geladen, die durch eine Vektornummer spezifiziert worden ist. Dies ist eine Nummer zwischen 0 und 15. Damit sind der Vektor und auch die Adresse festgelegt, auf die der Vektor zeigt, an dem die Exception beginnt. Die Vektoren liegen an den Adressen $80 bis $BF. Dies sind die Vektoren $20 bis $2F.

Zum Beispiel verzweigt der TRAP #1 zu der Adresse, die im Speicher an Adresse $84 (Langwort) steht.

Syntax	Flags XNZVC	Marke,(ea),n
Bcc Marke	- - - - -	Offset 8 oder 16 Bit
DBcc Dn,Marke	- - - - -	Offset 8 oder 16 Bit
BRA Marke	- - - - -	Offset 8 oder 16 Bit
BSR Marke	- - - - -	Offset 8 oder 16 Bit
JMP (ea)	- - - - -	(An),d(An),d(An,Rx),abs,PCR
JSR (ea)	- - - - -	(An),d(An),d(An,Rx),abs,PCR
RTE	★ ★ ★ ★ ★	wird gesetzt
RTR	★ ★ ★ ★ ★	wird gesetzt
RTS	- - - - -	
TRAP #n	- - - - -	0 bis 15

Das Programm

Nun kommen wir zu dem Programm dieser Ausgabe. Dieses Programm soll Ihnen einige grundlegende Arbeiten der Programmierung zeigen. Zu diesen Arbeiten gehört das Planen der Unterprogramme sowie deren Aufbau. Durch die Verwendung der Unterprogramme wird das Programm kürzer und übersichtlicher.

Das Programm soll zwei positive Dezimalzahlen addieren. Als erstes geben wir dazu eine Information aus (Einleitung). Dann soll die erste Zahl eingegeben werden. Dies machen wir, indem wir den Benutzer dazu auffordern. Ebenso geschieht dies mit der zweiten Zahl. Danach wird das Ergebnis, mit einem Text versehen, ausgegeben.

Die ersten beiden Texte auszugeben, dürfte Ihnen keine Schwierigkeiten bereiten. Danach muß man dem Benutzer die Eingabe ermöglichen. Dies erledigt eine Betriebssystemroutine (RLINE) für uns. Da wir diese Routine noch einmal benötigen, gestalten wir sie als Unterprogramm. Als Parameter benötigt diese Routine die Adresse eines Puffers. Der Puffer ist folgendermaßen aufgebaut: Das erste Byte des Puffers enthält vor Aufruf die maximale Anzahl der Zeichen, die eingelesen werden sollen. In dem zweiten Byte erhält man die tatsächliche Anzahl. Ab dem dritten Byte stehen die Zeichen. Die Eingabe wird durch die maximale Länge oder RETURN beendet, wobei RETURN nicht mit übergeben wird.

Da die Funktion RLINE Zeichen einliest, muß man noch überprüfen, ob nur Zahlen eingegeben worden sind. Die Umrechnung einer ASCII-Zahl in eine Binärzahl, ist recht einfach (ASCII-$30=Bin). Die stellenrichtige Zusammenfassung der Binärziffern erfolgt mittels dem Hornerschema (Siehe Ausgabe 9 „So rechnen Computer"). Die Testroutine wird, da sie zweimal benötigt wird, ebenfalls als Unterprogramm ausgeführt. Sie muß, bei Auftreten eines Fehlers, dies entsprechend kenntlich machen. Erstens durch Ausgabe eines Textes, und zweitens in einem Register, damit das Programm darauf reagieren kann.

Die Addition der beiden Zahlen erfolgt binär, und zwar mit dem ADD Befehl. Die Umrechnung des Ergebnisses in Dezimalziffern geht in genau der umgekehrten Reihenfolge. Da der DIVU Befehl maximal ein 16 Bit Ergebnis liefert, muß, damit kein Fehler auftritt, der Puffer entsprechend lang sein. Daraus folgt: 16 Bit = 65 535; Multipliziert mit zehn gleich 655 350; dann durch zwei geteilt ergibt 327 675. Dies entspricht der größten Zahl für jede Eingabe. Da die Eingabe auf die Anzahl der Ziffern beschränkt wird, folgt daraus: 5 Ziffern. Um dieses Programm leistungsfähiger zu machen, müßte man eine bessere Umwandlungsroutine entwickeln oder vor der Umwandlung auf die größte Zahl testen.

Natürlich wäre es auch möglich gewesen, die Eingabe in einer Zeile, mit einem + Zeichen zu trennen.

Dieses Programm wurde mit dem ST-Assembler geschrieben.

Sven Schuler

①

```
                        ; Additionsprogramm
                        ; Definitionen
conin equ 1
pline equ 9
rline equ 10
                        ; Programmstart
start                   ; Begrüßungstext ausgeben
 move.l #btxt,-(a7)
 move.w #pline,-(a7)
 trap #1
 addq.l #6,a7
einszahl                ; zur Eingabe auffordern
 move.l #b1txt,-(a7)
 move.w #pline,-(a7)
 trap #1
 addq.l #6,a7
 jsr eingabe            ; maximal 5 Zeilchen einlesen
 jsr test               ; nur Zahlen?
 cmp.l #-1,d0           ; Fehler aufgetreten
 beq einszahl           ; noch mal zur Eingabe
 jsr umrechnung         ; richtig, dann umrechnen
 move.l d0,d7           ; zwischenspeichern
zweizahl                ; Text ausgeben
 move.l #b2txt,-(a7)
 move.w #pline,-(a7)
 trap #1
 addq.l #6,a7
 jsr eingabe            ; maximal 5 Zeichen
 jsr test               ; nur Zahlen
 cmp.l #-1,d0           ; Fehler aufgetreten?
 beq zweizahl           ; noch ein mal
 jsr umrechnung         ; ansonsten umrechnen
 add.l d0,d7            ; und dazuaddieren
 move.l #austxt,-(a7)   ; Ausgabetext ausgeben
 move.w #pline,-(a7)
 trap #1
 addq.l #6,a7
 jsr bindez             ; d7 nach ASCII Dez und ausgeben
 move.w #conin,-(a7)    ; auf Taste warten
 trap #1
 addq.l #2,a7
 clr.w -(a7)            ; Ende des Hauptprogramms
 trap #1
                        ; Unterprogramme
eingabe                 ; feste Parameter
 move.l #puffer,-(a7)
 move.w #rline,-(a7)
 trap #1
 addq.l #6,a7
 rts
test                    ; testet den Puffer auf unerlaubte Zeichen
                        ; Ausgang: d0
 movem.l a1/d1,-(a7)    ; Register retten
 move.b puffer+1,d1     ; Anzahl der tatsächlichen Zeichen
 cmp.b #0,d1            ; Kein Zeichen?
 beq fehler             ; Fehlerbehandlung
 move.l #puffer+2,a1    ; Adresse des ersten Zeichens
```

②

```
tloop
 move.b (a1)+,d0        ; Zeichen holen
 cmp.b #'0,d0           ; mit 0 vergleichen
 blt fehler1            ; kleiner? dann Fehler
 cmp.b #'9,d0           ; mit 9 vergleichen
 bgt fehler1            ; größer? dann Fehler

 sub.b #1,d1            ; nächstes Zeichen
 bne tloop              ; noch welche, ansonsten weiter
 movem.l (a7)+,a1/d1    ; Register wieder laden
 rts                    ; zurück
umrechnung
                        ; Ausgang: d0
 movem.l a1/d1/d2,-(a7) ; Register speichern
 move.l #puffer+2,a1    ; Adresse des ersten Zeichens
 move.b puffer+1,d1     ; Anzahl der Zeichen
 clr d0                 ; Anfangswert =0
uloop
 mulu #10,d0            ; mit 10 multiplizieren
 move.b (a1)+,d2        ; Zeichen holen
 and.b #$0f,d2          ; oberes nibble ausblenden (-$30)
 add.l d2,d0            ; aufaddieren
 sub.b #1,d1            ; nächstes Zeichen
 bne uloop              ; noch? ansonsten weiter
 movem.l (a7)+,a1/d1/d2 ; Register wieder laden
 rts                    ; ende
fehler                  ; ftxt ausgeben
 move.l #ftxt,-(a7)
 move.w #pline,-(a7)
 trap #1
 addq.l #6,a7
 bra aus
fehler1                 ; f1txt ausgeben
 move.l #f1txt,-(a7)
 move.w #pline,-(a7)
 trap #1,a7
 addq.l #6,a7
aus
 move.l #-1,d0          ; Fehler aufgetreten setzen
 movem.l (a7)+,a1/d1    ; Register retten
 rts                    ; ende
bindez                  ; bin nach dez mit Ausgabe
                        ; Eingang: d7
 movem.l a2/d7,-(a7)    ; Register retten
 movea.l #puffer+7,a2   ; Pufferadresse für letzten Zeichen
dloop
 divu #10,d7            ; durch 10
 swap d7                ; Rest ins Lowword
 add.b #$30,d7          ; +$30 ASCII
 move.b d7,-(a2)        ; im Puffer ablegen
 clr.w d7               ; Rest löschen
 swap d7                ; wieder tauschen
 cmp.w #0,d7            ; noch was da?
 bne dloop              ; dann weiter
```

Signum!



bald erhältlich

Fraktur ift drin, Σ ∞ ∫ № Π ist drin, *Palette ist drin.* **Normande ist drin**, *kursiv ist drin.* αβχδεφ ist drin, natürlich proportional, hervorragend auf dem NEC P6 !

Signum! ist für Literaten, Wissenschaftler, für Studenten und für alle, die Schrift benutzen.

448,— DM

Brückenstraße 47, 6900 Heidelberg,
Telefon 06221/410134

APPLICATION SYSTEMS HEIDELBERG

ST-SOFTWARE

Ja, da sind wir...
ST-Spezial-Versand

Hard- & Software für ATARI-ST's

Keine Beratung – nur Verkauf
dadurch enorm **PREIS-wert**

Beispiele:

- Art director **198,–**
- Film director **225,–**
- Mac Emulator **498,–**

am Lager

Spar-Preisliste anfordern

M. Diesenberger
Westpreussenring 26
2400 Lübeck 14

Tel: 0451-302830



PRG für ATARI ST – EXZELLENT IN STRUKTUR, GRAFIK, SOUND – ALLE PRG IN DEUTSCH – ALLE PRG IN S/W UND FARBE

GELD – 25 Rechenroutinen mit Ausdruck für Anlage – Sparen – Vermögensbildung – Amortisation – Zinsen (Effektiv-/Nominal-) – Diskontierung – Konvertierung – Kredit – Zahlungsplan usw. 96,–

GESCHÄFT – Bestellung, Auftragsbestätigung, Rechnung, Lieferschein, Mahnung 6 Briefrahmen mit Firmendaten zur ständigen Verfügung (Anschrift, Konten usw.. Menge/Preis, Rabatt/Aufschlag, MWst., Skonto, Verpackung, Versandweg usw.) 96,–

ETIKETTENDRUCK 'bedruckt 40 gängige Computer-Haftetiketten-Formate nach Wahl und Auflagebestimmung, kinderleichte Gestaltung, Ablage für wiederholten Gebrauch 89,–

ASTROLOGISCHES KOSMOGRAMM – Nach Eingabe von Namen, Geb. Ort (geogr. Lage) und Zeit werden errechnet: Siderische Zeit, Aszendent, Medium Coeli, Planetenstände im Zodiak, Häuser nach Dr. Koch/Schäck (Horoskop-Daten m. Ephemeriden) – Auch Ausdruck auf 2* DIN A4 mit allgemeinem Persönlichkeitsbild und Partnerschaft 75,–

BACKGAMMON – überragende Grafik, gänzlich mausgesteuert, ausführliche Spielanleitung, lehrreiche Strategie des Computers, in 6 Farben bzw. Grauabstufungen bei S/W 58,–

CASINO-Roulett – Mit Schnellsimulation, Chancentest, Sequenzenverfolgung, Kassenführung 68,–

KALORIEN-POLIZEI – Nach Eingabe von Größe, Gewicht, Geschlecht, Arbeitsleistung erfolgt Bedarfsrechnung und Vergleich m. d. tatsächlichen Ernährung (Fett, Eiweiß, Kohlehydrate), Idealgewicht, Vitalstoffe, auf Wunsch Ausdruck 56,–

BIORHYTHMUS – zur Trendbestimmung des seelisch-/geistig-/körperlichen Gleichgewichts, Zeitraum bestimmbar – Ausdruck mit ausführlicher Beschreibung über beliebigen Zeitraum 56,–

FONT EDITOR unter DEGAS: 12 bekannte Schriftarten m. deutschem Zeichensatz 64,–

Bibliothek – schreibt Listen oder Etiketten, auch auszugsweise nach codierten Kriterien (Titel/Best.-Nr./Verfasser/Verlag/Gruppe/Bestand/Preis/usw.) 86,–

GEM + MERGE: Die GEM (AES + VDI)-Applikationen unter ST Basic – ca. 100 Routinen zum Mergen für Funktion und Aufruf – Führt zur GEM-Beherrschung 58,–

usw. usw. – Fordern Sie mit Freiumschlag unsere Liste an!
Im Computer-Center oder bei uns zu obigen, unverbindlich empfohlenen Preisen + DM 3,– bei Vorkasse oder DM 4,70 bei Nachnahme.

I. DINKLER, IDEE-SOFT, Am Schneiderhaus 7
D-5760 Arnsberg 1 · Tel.: 02932/32947

ATARI ST

Spitzen C-Entwicklungspaket
Mark Williams C-Compiler 498,–

Qualitätssoftware METACOMCO
Lattice C deutsche Anleitung 348,–
MCC Pascal ISO Standard 248,–
Cambridge LISP Metacomco 490,–
MCC Assembler Metacomco 168,–

Spitzensoftware PROSPERO
Pro-Pascal ANSI/FIPS/ISO/BSI 448,–
Pro-Fortran77 ANSI X3.9-1978 490,–

LDW-Basic Compiler 159,–
K-Spread Tabellenkalkulation deutsch ... 168,–
K-Graph Grafik zu K-Spread 118,–
K-Switch Programmumschalter 118,–
K-Comm Terminalprogramm VT100 148,–
K-Resource Construction Set 118,–

Art Director Grafikprogramm 178,–
Film Director Animation 198,–

Pawn Text-Abenteuer 68,–
Psion Chess Schachspiel 3D 78,–
ST Karate Sportsimulation 75,–
Arena Sportsimulation 78,–
Deep Space Weltraumspiel 88,–
Flight II von Sublogic 148,–

Diskettenlaufwerke Firma CUMANA
Doppellaufwerk 3,5" 720 KB 1090,–
5 1/4 Diskettenlaufwerk 40/80 690,–
5 1/4 und 3 1/2 Doppellaufwerke 1190,–
OS-9 Betriebssystem Info anfordern
Harddisk 10 MB Bausatz 1198,–
Harddisk Back-up Programm 78,–
Harddisk 20 MB 8 Partitions 2198,–

Preisliste mit Info anfordern.
Händleranfragen erwünscht!

PHILGERMA GmbH
Ungererstraße 42 · 8000 München 40
Tel. (089) 395551 ab 10 Uhr

NEU
GRAFIKKARTE für ATARI ST

1024 x 1024 Pixel Auflösung, 128 K Byte RAM
Grafikprozessor zieht hardwaremäßig mit 1,5 Mill. Pixel/sec
Punkte, Linien, Kreise, Rechtecke.
Schrift: normal & italic. Scrollen, Zoom
Kein Eingriff in den Rechner!
incl. Netzteil + Software
Gehäuse: Slim-Line als Monitorständer

nur 698,–
(Einführungspreis)

Ulrich Breuer & Andreé Benninghoff
Sonnenstr. 24 · 5800 HAGEN 7
Tel: 02331/406973

(3)

```
ausgabe
  move.l a2,-(a7)          ; ab letztem Zeichen ausgeben
  move.w #pline,-(a7)
  trap #1
  addq.l #6,a7

  movem.l (a7)+,a2/d7      ; Register wieder laden
  rts
  .data                    ; Datenbereich
btxt
  dc.b 27,"E",27,"e"
  dc.b "Dieses Programm addiert zwei positive Zahlen miteinander."
  dc.b 10,10,10,13,0
  .even
b1txt
  dc.b 10,13,"Bitte die erste Zahl",10,13,0
  .even

b2txt
  dc.b 10,13,"Bitte die Zweite Zahl",10,13,0
  .even
ftxt
  dc.b 10,13,"Bitte mindestens eine!!!",10,13,0
  .even
f1txt
  dc.b 10,13,"Bitte Zahlen!!!",10,13,0
  .even
austxt
  dc.b 10,13,"Das Ergebnis ist "
  .even
puffer
  dc.b 5,0
  dc.b 0,0,0,0,0,0
  .even
  .BSS ; Ende des Datenbereichs
```

Der Assembler zum Kurs

Alle abgebildeten Beispiele dieses Assembler Kurses, wurden mit dem ST-Assembler erstellt. Dieser Assembler wurde uns von Eckhart Kruse zur Verfügung gestellt und ist auf der Public-Domain Diskette Nr. 34 erhältlich.

Der Autor schrieb damit übrigens das Siegerprogramm des ST-Musik Wettbewerbs.

Die ST-Computer Redaktion liefert damit das notwendige Werkzeug zum praktischen Einstieg in die Assemblersprache. Das Programm läuft auf mittlerer (Farbe) und hoher Auflösung (s/w).

Wer keinen handelsüblichen Assembler besitzt, dem sei damit geholfen.

SIGNUM!

Textverarbeitung in einer anderen Dimension

SIGNUM verfolgt ein völlig neues Konzept bei der Textverarbeitung. Es arbeitet ausschließlich im Grafikmodus und bietet dadurch ungeahnte Möglichkeiten. Der Zeichensatz kann frei definiert werden und die einzelnen Zeichen sind pixelweise positionierbar. Der Clou ist aber, daß Blocksatz und Proportionalschrift auf dem Bildschirm dargestellt werden und dies genau dem späteren Ausdruck auf dem Drucker entspricht.
Das klingt vielleicht alles sehr utopisch, doch das Kunststück ist wirklich gelungen. Die Druckqualität eines 24-Nadel-Druckers erinnert somit sehr an die eines teuren Laserdruckers.

Neben den Funktionen, die man von einem Textverarbeitungsprogramm erwartet, verfügt SIGNUM noch über einige ungewöhnliche Features. Sieben(!) Zeichensätze können gleichzeitig in einem Dokument verwendet werden. Dabei sind drei direkt erreichbar (normal mit 'Alternate', mit 'Control'), die restlichen sind über Escape-Sequenzen aufrufbar.

Da die gesamte Textverarbeitung im **Graphikmodus** abläuft, gibt es bei Signum keine Beschränkung bei der Positionierung der Zeichen. Nicht Zeilen und Spalten bestimmen das Schriftbild, sondern der Benutzer kann jeden Buchstaben frei plazieren. Somit ist das Schreiben von naturwissenschaftlichen Formeln und Brüchen ein Kinderspiel. Diese Anwendung war auch der Anlaß für Franz Schmerbeck, den Autor von SIGNUM, ein neues Textverarbeitungskonzept zu entwickeln.

Zusammen mit dem komfortablen Zeichensatzeditor kann außerdem jedes beliebige noch so exotische Zeichen erstellt werden; auch für eigene Kreationen ist der Weg offen. Wie wäre es zum Beispiel mit einem kleinen, aber persönlichen Emblem oder Wappen als Briefkopf?

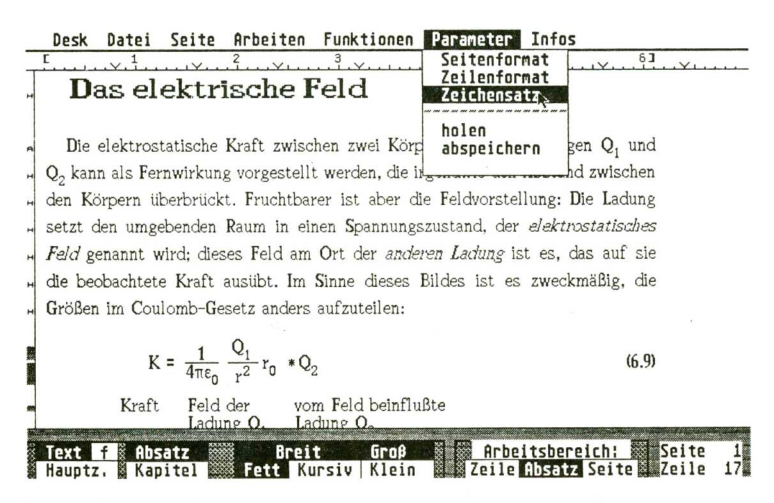


Bild 1: Textverarbeitung

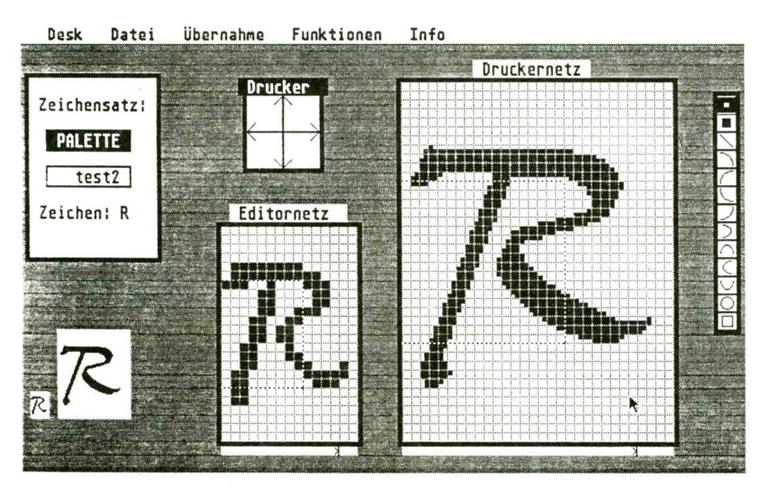


Bild 2: Zeichensatzeditor

Das Editieren auf dem Bilsdchirm ist teilweise etwas gewöhnungsbedürftig. Dies ist jedoch in erster Linie auf die vielfältigen Möglichkeiten zurückzuführen, die dem Anwender zur Verfügung stehen. Der Cursor ist schnell, mittels Maus auf dem gesamten Bildschirm positionierbar. Das Scrollen (Blättern) des Textes ist ebenfalls mit der Maus möglich und in diese Art sehr einfach und vor allem erfreulich schnell. Dazu fährt man, bei gedrückter rechter Maustaste, an den Bildschirmrand – und der Text bewegt sich gleitend in die gewünschte

ZEIT FÜRS AUT(

Gerd Karpe gibt mal wieder

Hier ein paar wichtige Ratschläge z

Waschen Sie Ihr Automobil mit hand nausen spritzen gedankenlos kaltes hochempfindliche Limousine. Kein die Startautomatik röchelt und der art unsachgemäßer Behandlung erlei greifenden Schock, wie jüngste Te bewiesen haben.

Reinigen Sie also Ihren PS-Gefähr großporigen Schwamm. Suchen Sie geschütztes Plätzchen. Die roh frischer Zugluft im Fr grundfalsch he diesen sc tar

Application Systems /// Heidelberg, Postfach 102646
6900 Heidelberg, ☎ 06221/410134

SIGNUM!
TEXTVERARBEITUNGSSYSTEM

Das Textverarbeitungssystem **Signum!** wurde speziell für die Computer der ST-Serie von Atari entwickelt. Es verwirklicht konsequent ein neues Konzept: Sämtliche Zeichen sind vom Benutzer frei definierbar. Dadu ... es möglich, auch Non-Standard-Texte in hoher Druckqualität zu erstellen.

Signum! ... ditoren, ... Pa- ...).

Susi Software
Atariplatz 45
1000 Berlin 45
28.09.1986

Herrn
Robert Huber
Willibaldring 17a
4711 Hintertupfing 80

Sehr geehrter Herr Huber,

zunächst möchten wir uns für Ihr Interesse an unserem Textverarbeit Edna bedanken.

Die beiliegende Diskette enthält eine Vorversion. Sie finden d lichen Texteditor Edna, den Druckertreiber Pr24n und den für 24-Nadeldrucker. Der Ordner Chsets enthält einige schiedene Schriftarten. In Dokus schließlich sind einige B die mit Edna erstellt wurden. Bitte erstellen Sie sich ei ...ten Diskette.

Klassiker Modern

Habe nun, ach! Philosophie,
Bierbrauerei und Medizin
Und leider auch Theologie
Durchaus studiert, mit heißem Bemühn.
Da steh ich nun, ich armer Tor,
Und bin so klug als wie zuvor;
Heiße Magister, Doktor gar,
Und ziehe schon an die zehen Jahr
Herauf, herab und quer und krumm
Meine Lehrer an der Nase herum =
Und sehe, daß wir nichts wissen können!
Das will mir schier das Herz verbrennen.

Zwar bin ich gescheiter als all die Laffen,
Doktoren, Politiker, Schreiber und Pfaffen;
Mich plagen keine Skrupel noch Zweifel,
Fürchte weder Mac Donalds noch Teufel =
Dafür ist mir auch alle Freud entrissen,
Bilde mir nicht ein, was Links zu wissen,
Bilde mir nicht ein, ich könnte was rühren,
Die Menschen zu bessern und zu verführen.

Auch hab ich weder Porsche noch Geld,
Noch Ehr und Herrlichkeit dieser Welt.
Es möchte kein Hund so länger leben!
Drum hab ich mich dem Computer ergeben,
Ob mir durch Elektrons Kraft und M...
Nicht manch Geheimnis wü...
Daß ich nicht meh...
Zu rech...

Bild 3: verschiedene Texte

Richtung.

Die Cursortasten bewegen den Cursor normalerweise in Zeichenabständen und vertikal auf den sogenannten 'Hauptzeilen'. Die Bewegungen können jedoch auf Minischritte (pixelweise) reduziert werden. Gleichzeitiges Drücken von Shift & Control bewegt den Cursor auch gemäß der verwendeten Proportionalschrift, damit wird das Editieren dieser Texte erst möglich.

Eine Unmenge von weiteren Tastenkombinationen erleichtert das Anspringen von Worten, Zeilen und Marken. Textattribute, wie z. B. Fett- oder Kursivschrift, werden entweder direkt mit der Maus angeklickt oder über **Escape-Sequenzen** ausgewählt.

Sehr praktisch ist der **Textpuffer**, der über Funktionstasten mit Textstücken aufgefüllt wird. Auf Tastendruck kann dieser Text an einer beliebigen Stelle (auch mehrfach) ausgegeben werden.

Die **Funktionstasten** sind vorbelegt und nicht änderbar. Dafür gibt es die Möglichkeit, beliebige Tasten mit Befehlen oder **Makros (Befehlssequenzen)** zu belegen. Dieser Vorgang ist sehr einfach, denn nach dem Anwählen der gewünschten Taste wird der von nun an eingegebene Text (auch Steuerzeichen) übernommen. Erst nach Beenden dieser Funktion ist die Taste aktivierbar.

Zum Bearbeiten von Textteilen stehen dem Anwender eine Reihe von **Blockoperationen** zur Auswahl. Der Blockbereich wird dazu mit der Maus angewählt und durch ein Raster kenntlich gemacht. Die S hriftart kann in einem solchen Feld einfach geändert werden, auch Operationen wie Löschen und Verschieben sind durchführbar. Dies ist vor allem zum Editieren von Formeln gedacht, falls diese nicht das gewünschte Aussehen haben.

Verschieben erfolgt mit den Cursortasten in großen oder kleinen Schritten (pixelweise), allerdings geht dabei der Hintergrund verloren. Hier wäre ein Bildschirmpuffer angebracht, der ein versehentliches Löschen rückgängig macht.

SIGNUM besitzt **Formatierbefehle**, die es erlauben einen geschriebenen Text

umzuformatieren. So kann ein Text mit Flatterrand in den schöneren Blocksatz überführt werden.

Eine weitere Option ist der Zeilenumbruch, der auf Wunsch automatisch durchgeführt wird. Es gibt aber auch eine halbautomatische Einstellung, die beim Erreichen eines über den rechten Rand gehenden Wortes anhält. Der Benutzer entscheidet dann mit einem einfachen Mausklicken, an welcher Stelle das Wort getrennt werden soll. Diese Methode ist zwar nicht so komfortabel wie eine vollautomatische Silbentrennung, sie hat aber den Vorteil, daß sie keine Fehler machen kann.

Bei SIGNUM gibt es einen gesonderten Menüpunkt für **Parameter**. Dort läßt sich z. B. das Seiten- und Zeilenformat einstellen. Dazu gehört auch die Lage der Kopf- und Fußzeilen. Da alle Angaben in Zoll eingegeben werden müssen, ist es erforderlich, die Maße umzurechnen. Durch Betätigen der Option 'Parameter abspeichern' werden diese Parameter und zusätzlich auch noch die gewählten Zeichensätze und die programmierten Tasten abgespeichert. Die Parameter werden beim Starten des Programms automatisch mitgeladen.

Dem **Zeichensatzeditor** ist ein eigener Abschnitt gewidmet, weil er sich nicht in einer Zeile abhandeln läßt. Außerdem ist er ein eigenständiges Programm, welches nicht von SIGNUM aufgerufen wird. Zum Erstellen eines Zeichens steht ein **Raster von 30 mal 40 Punkten** zur Verfügung. Diese Auflösung ist höher, als sie auf dem Bilschirm (15x24) dargestellt wird, aber ein 24-Nadeldrucker kann damit problemlos arbeiten, so erklärt sich die hohe Qualität des Ausdrucks.

Die Zeichen können im Editornetz oder im Druckernetz erstellt werden, wobei sich das Druckernetz, wegen der besseren Auflösung, anbietet. Durch einen einfachen Mausklick werden sie dann in das jeweils andere Netz übernommen.

Zum Entwerfen von Zeichen stehen umfangreiche Funktionen zur Verfügung, neben Linien, Kreisen und Bögen, gibt es Funktionen zum Verschieben, Spiegeln und Strecken.

Zum Schluß noch ein Wort zum **Ausdruckvorgang** an sich, denn der hat auch so seine Besonderheiten. Jede Zeile wird bei einem 24-Nadeldrucker in ca. zwei Durchgängen erstellt, je nach Höhe des Zeichensatzes. Bei einem 8- bzw. 9-Nadeldrucker wird jede Zeile sogar sechsmal abgefahren, um die erwünschte Qualität zu erreichen. Dies wirkt sich, wie man sich denken kann, direkt auf die Geschwindigkeit aus, mit der ein Ausdruck erfolgen kann. So braucht z. B. der 24-Nadeldrucker von NEC (P6), je nach eingestellter Druckqualität, ca. eine bis vier Minuten für eine DIN A4 Seite, ein 8- bzw. 9-Nadeldrucker, entsprechend seiner Geschwindigkeit und dem dreimal häufigeren Durchlauf, erheblich länger. Da jedoch das Aussehen des Ausdrucks mit dem des Bildschirm identisch ist, entfallen die Kontrollausgaben völlig. Außerdem wird die erreichte Qualität des Ausdrucks mit Sicherheit den Benutzer über die etwas längere Wartezeit hinwegtrösten.

SIGNUM kann **ASCII-Dateien** einladen und versieht sie dabei mit einem beliebigen Zeichensatz. Das Abspeichern im ASCII-Format ist jedoch nicht möglich. Somit ist das Programm zum Erstellen von Source-Code ungeeignet und auch Korrekturprogramme wie z. B. 1ST-LEKTOR können nicht eingesetzt werden. Eine Option für MAIL MERGE oder das Einbinden von Graphiken fehlt ebenfalls. Wie mir Herr Schmerbeck versicherte, wird die eine oder andere Funktion noch realisiert werden und in Form eines Updates für den Käufer erhältlich sein.

Zum Arbeiten mit SIGNUM ist mindestens eine ATARI ST mit 512 K RAM und dem TOS auf ROMs notwendig. Auf dieser Konfiguration können dann jedoch nur ca. drei Seiten editiert werden. Ein vernünftiges Arbeiten erfordert deshalb **mindestens einen 1 MB Rechner**, am besten ebenfalls mit dem TOS auf ROMs, falls man mit einer größeren RAM-Disk arbeiten will.

Ein Vergleich mit dem weitverbreiteten 1ST-WORD ist eigentlich nicht möglich, da die Zielgruppe eine andere ist. Während 1ST-WORD eher ein Programm zum einfachen und schnellen Erstellen von Texten ist, gehört SIGNUM in die gehobenere Kategorie. Es ist für Anwender gedacht, die Funktionen zum freien Positionieren von Zeichen benötigen und Interesse oder Bedarf an speziellen Zeichensätzen haben. Natürlich ist es damit auch für jene von unschätzbarem Wert, die den Text in seiner endgültigen Form (mit Blocksatz und Proportionalschrift) schon auf dem Bildschirm sehen wollen, um sich mehrmaliges Ausdrucken zu ersparen.

SIGNUM bietet zum Preis von DM 445,– ein außergewöhnliches Textverarbeitungsprogramm, einen komfortablen Zeichensatzeditor, verschiedene (komplette) Zeichensätze und Druckertreiber für 24- und 8/9-Nadeldrucker. Angesichts der gebotenen Leistungen ein faires Angebot und in seiner Art völlig konkurrenzlos. (mn)

Vertrieb:
Application Systems /// Heidelberg
Postfach 10 26 46
6900 Heidelberg

Die **Neuen** und die **Erfolgreichen aus dem Heim-Verlag**



Wenn Sie das Software-Paket **VIP-Professional** kaufen wollen oder schon besitzen, dann weiht Sie dieses Buch schnell und umfassend in die Geheimnisse dieses Profiprogrammes ein.

VIP-Professional besteht aus den drei Funktionsbereichen
- DATENBANK
- KALKULATION
- GRAFIK

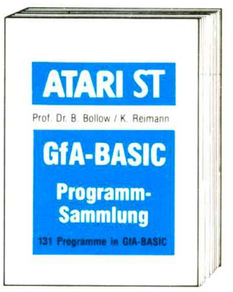
mit denen wichtige und vielfältige Aufgaben hervorragend gelöst werden.

Das Buch enthält komplette Musterlösungen für die Gewinn- u. Verlustrechnung und Fakturierung. Anhand dieser Beispiele wird gezeigt, wie Sie alles herausholen, was in VIP-Professional steckt.

Mit diesem Buch können Sie **VIP-Professional** richtig einsetzen und seine Möglichkeiten voll ausschöpfen.

B-408 Buch 49,– DM
D-428 Diskette 39,– DM

Anhand von über 130 Programmbeispielen und Erläuterungen lernen und trainieren Sie das Programmieren in BASIC. Von einfachen, aber grundlegenden Beisp. bis zur ausgereiften Anwendung findet der ATARI ST-Besitzer Beispiel-Programme, die den perf. Einstieg in die Programmiersprache des GfA-BASIC's leicht machen. Alle Beisp. werden so dargestellt, daß das Verständnis für die Programmstruktur gefördert wird, aber auch die Details der GfA-BASIC-Version deutlich werden. Die Progr. sind gut dokumentiert, wobei die hervorr. Möglichkeiten des GfA-Basic zur strukturierten Programmierung genutzt werden. **Aus dem Inhalt:** ● Eigenschaften des GfA-BASIC ● Dateiverwalt. u. Ordner in GfA-BASIC ● Schleifentechnik ● Felder ● Unterprogramme ● Menü-Steuerung ● Window Technik ● Zufallszahlen ● Seq.- u. RANDOM-Dateien ● Textverarbeitung ● Sortierprogramme ● Fakturiersysteme ● Grafik ● Spiel uvm.
Zum Buch: Programmdiskette mit sämtl. Beispielprogr. 320 Seiten mit 131 Programmen in GfA-BASIC



B-407 Buch 49,– DM
D-427 Diskette 49,– DM

Das Buch für den richtigen Einstieg mit dem ATARI ST. Leicht verständlich wird der Lernende in den Lernstoff eingeführt.

Einige der Themen:

Die Hardware des ATARI ST
- Überblick über die Systemkomponenten und ihre Funktionsweise
- Aufstellung und Wartung des Computers

Die Software des ATARI ST
- Arbeiten mit dem GEM (das Desktop/Maus/Icons etc.)
- Kopieren von Files und Disketten, Löschen und Formatieren u.v.a.m.
- Die Programmiersprachen BASIC und LOGO
- Einführung in die Bedienung und Programmierung
- Programmsammlung mit vielen interessanten Beispielen

350 Seiten mit 55 Abbildungen.



B-400 Buch 49,– DM
D-420 Diskette 39,– DM



B-402 Buch 49,– DM
D-422 Diskette 39,– DM

Der ideale Einstieg in die Programmierung mit LOGO. Ein Buch für alle, die LOGO schnell verstehen und perfekt erlernen möchten. Ein Buch wie es sein muß, leicht zu verstehen und interessant geschrieben.

Einige der Themen:
- **Variablen- und Listenverarbeitung**
- **Packages und Properties**
- **Die Turtlegrafik**
- **Viele ausführlich erklärte Beispielprogramme zu den Themen**
  - Arbeiten mit relativen und sequentiellen Dateien
  - Anwendungen aus Mathematik
  - Hobby- und Spielprogramme u.v.a.m.

Zum Buch gibt es die Programmdiskette mit sämtlichen Beispielprogrammen.

ca. 350 Seiten mit vielen Abbildungen.

Ein Grundwerk für alle ATARI ST-Besitzer. Auf über 300 Seiten eine klare und verständliche Einführung in die Programmiersprache BASIC, elementare BASIC-Kommandos, Diskettenhandhabung und vieles, was zur perfekten Beherrschung des ATARI ST gehört.

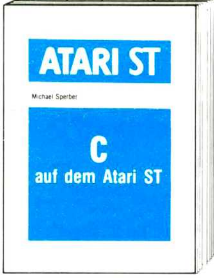
Ein Spitzenbuch mit über 80 Übungs- und Anwenderprogrammen wie z. B.:

Sortierprogramme / Textverarbeitung / Umgang mit sequentiellen- u. Random-Dateien / Fakturierprogramm / Programmiertechniken an ausgewählten Beispielen u. v. a. m.

Zum Buch gibt es die Programmdiskette mit sämtlichen Beispiel-Programmen.



B-404 Buch 49,– DM
D-424 Diskette 39,– DM



B-406 Buch 49,– DM
D-426 Diskette 39,– DM

C ist die zweite „Muttersprache" des Atari ST: schnell, komfortabel, kompakt im Code.

„C auf dem Atari ST" ist für Anwender geeignet, die Erfahrungen mit anderen Programmiersprachen gemacht haben. Das Buch behandelt den vollen Sprachumfang von Standard-C und verweist auf BASIC und Pascal.

Besonderer Wert wird auf die Anschaulichkeit und Genauigkeit der Darstellung gelegt. Alle Programmbeispiele sind getestet und direkt in den Text übernommen.

Aus dem Inhalt: C-Compiler für den Atari: Digital Research, Lattice, Megamax ● der Editor ● Bedienung des Compilers ● Grundlegende Elemente eines C-Programms ● Variablentypen ● Felder und Vektoren ● Ausdrücke ● Zeiger ● Speicherklassen ● Bitfelder ● Varianten ● Aufzählungen ● Dateien ● Diskettenhandling ● Einbindung von Assemblerprogrammen ● Bildschirmgrafik in C ● Fehler in den C-Compilern ● Tools u. a.

Ein Grundwerk.
Wer sich mit der Programmierung der GEM-Funktionen vertraut machen will braucht dieses Buch!
Es beginnt mit einer Erläuterung des GEM-Aufbaus und führt anhand von Programmbeispielen zum **leichten Verständnis aller auf dem Atari verfügbaren GEM-Funktionen.** Die übersichtliche Gliederung ermöglicht auch ein schnelles Nachschlagen der Funktionen.

Einige der Themen:

● Was ist **GEM** ● Die GEM-Bestandteile **VDI** und **AES** ● Die **GEM-Implementation** auf dem Atari ST ● Aufruf der GEM-Funktionen aus **BASIC, C,** und **ASSEMBLER** ● Sonstige **Programmiersprachen** und GEM ● Die Programmierung der **VDI**-Funktionen ● Die **AES**-Bibliothek und die Programmierung ihrer Funktionen ● Aufbau eines **Objektbaumes** ● Was ist eine **Resource**-Datei? ● Viele erläuterte **Beispielprogramme** in BASIC, C und ASSEMBLER.



B-401 Buch 49,– DM
D-421 Diskette 39,– DM



B-405 Buch 49,– DM
D-425 Diskette 39,– DM

Zum neuen Basic Interpreter, ein Buch, das mit gezielten Beispielen verständlich den Einstieg in das Basic der Superlative ermöglicht. Ein Muß für jeden Besitzer dieses Interpreters. Einige der Themen:
- Der Umgang mit dem Editor
- Ausführliche, mit Beisp. versehene Befehlsübersicht
- Die fantastischen Grafikmöglichkeiten (Windows, Sprites, Alertbox, Pull-down Menüs
- Strukturierte Programmierung – auch in Basic möglich!
- Dateiverwaltung unter GfA-Basic
- GEM Handhabung in Basic
- Vergleich mt anderen Basic-Interpretern – Hilft bei der Kaufentscheidung
- Zahlreiche Übungs- und Anwenderbeispiele
- Mathematik und hohe Genauigkeit

Zum Buch gibt es die Programmdiskette mit sämtlichen Beispielprogrammen.

ACHTUNG: Zu den Büchern gibt es die Programmdiskette mit allen Programmbeispielen – Diskette laden und los geht's...



6100 Darmstadt-Eberstadt
Heidelberger Landstr. 194
Telefon 0 61 51 / 5 60 57

▶ **Für Bestellungen verwenden Sie am besten die in der ST-Zeitschrift vorhandene BUCH- UND SOFTWARE-BESTELLKARTE**
▶ **Bücher und Programm-Disketten aus dem Heim-Verlag erhalten Sie auch bei Ihrem ATARI-Fachhändler oder im Buchhandel**

*alle Preise sind unverbindliche Preisempfehlungen

Public-Domain Service

Public-Domain Software hat sich bewährt. Sie ermöglicht dem ST-Benutzer den Zugang zu guter Software ohne gleich den Geldbeutel zu belasten.
Viele Programme haben durchaus 'professionelle' Eigenschaften und dies zum Nulltarif. Das momentane Angebot kann sich sehen lassen. Malprogramme, Diskmonitore, div. Utilities, Desk-Accessories, Dateiverwaltung, Terminalprogramm und sogar Programmiersprachen wie Forth, LISP oder Prolog 10. Auch einige wirklich schöne Spiele sind momentan verfügbar.
Da das Kopieren dieser Software bei Händlern nicht möglich ist, haben wir diesen Kopierservice eingerichtet.
Für nur 5,– DM Unkostenbeitrag pro Diskette für das Kopieren und Bearbeiten, bieten wir Ihnen folgende Public Domain Programme an:

1. Diskette
● *Neochrome V.06:* Malprogramm [f] ● *Doodle:* Malprogramm [s/w]

2. Diskette
● *Joshua:* Diskmonitor – CP/M Emulator (CP/M 80) ● *Megaroids:* Weltraumspiel ● *Diskformatter:* erweitert Diskkapazität auf 399 K bzw. 809 K ● *Kartei-Kasten:* luxuriöses Karteiprogramm ● *50/60 Hz*-Umschaltung für Monitore.

Diskette 3 und 4 beinhalten ein komplettes *FORTH* System *(volksFORTH-83)*. Das System umfaßt einen Screen-Editor, Interpreter, Compiler und diverse Utilities.

5. Diskette
● *Ramdisk* mit Source-Code ● *SG 10* Druckertreiber als Desk-Accessory ● *JAM Term:* Terminalprogramm ● *Diskcopy* ● *ST Grafic:* bewegte Grafiken in 3D ● *CP/M Emulator V 8.4:* neuste Version

6. Diskette
Diese Diskette beinhaltet einige Nutzprogramme für den CP/M-Emulator. Lauffähig mit Version 8.4.
● *CPMTOS:* TOSCPM wandelt CP/M-ASCII-Files in TOS-Format und zurück ● *SCOPY:* Kopierprogramm für ein Laufwerk ● *DDTZ:* Debugger ● *SETKEY:* Tastaturbelegung, und vieles mehr!!

7. Diskette
Die Sprache *LISP* (künstliche Intelligenz) ist nun auch auf Public-Domain erhältlich. (incl. 54 kByte Dokumentation, Anleitung und Beispielen)

8. Diskette
● *Diskmon:* ● *GfA Basic-Demo:* ● *Kuvert:* zum Bedrucken von Kuverts ● *Mini-CAD:* ein praktisches Programm mit Source-Code ● *Vokabel:* Vokabeltrainer ● *Superformat:* unterschiedliche Diskformate

9. Diskette
● *Funkplot:* Funktionsplotprogramm incl. Funktionseingabe über Tastatur, Integration und Differenzation ● *Konvert:* Konvertierung von Farb- auf Monochrombilder in 3,6 Sekunden ● *Spriteed:* ein kleiner Spriteeditor ● *Super:* ein Superhirnspiel [f]

10. Diskette
● *Iconeditor* ● *Puzzlepuzzle:* Puzzleprogramm [s/w] ● *Krabat-Schach* Schach für den ST [s/w]

11. Diskette
Auf dieser Diskette bieten wir eine weitere Programmiersprache an. Es handelt sich hierbei um PROLOG 10. Diese 'intelligente' Sprache unterstützt zusätzlich die GEM-Routinen. Auch mit der Anleitung wird nicht gegeizt, denn auf der Diskette befinden sich ca. 95 KByte Anleitung.

12. Diskette
● *Rechner:* ein luxuriöser Taschenrechner ● *Calc:* Taschenrechner als Desk-Accessory ● *Biorhythmus:* Erläuterung überflüssig ● *Hypnose:* eine besondere Zugabe

13. Diskette
Diese Diskette enthält einige ganz besondere Leckerbissen. ● *Pascalshell:* Menüsteuerung in GEM für ST-Pascal. ● *MMCOPY .ACC:* Filekopierprogramm als Accessoir ● *Happy 4:* 'Vier gewinnt' ● *Labyrint . ACC:* Labyrint als Accessoir ● *MAD . ACC:* Ihr Rechner entwickelt Eigenleben ● u. a.

14. Diskette
● *Tempelmon:* Maschinensprachemonitor von Th. Tempelmann [s/w] ● *Transistorvergleicher:* Verwaltung von Transistordaten ● *Bitte ein Bit:* Grafisch unterlegtes Bitcopyprogramm ● *Diskmon:* Diskmonitor auch für doppelseitige Laufwerke ● *VIFibu:* Kleine Finanzbuchhaltung aus Frankreich (Merci bien) ● *EX 06:* Fängt ca. 20 % der Systemfehler auf und gibt Fehlerart an

15. Diskette
Druckeranpassung 1st_Word
Auf dieser Diskette befinden sich alle uns zur Verfügung stehenden Druckertreiber. Wir aktualisieren diese Diskette ständig. Auch Sie können Ihre Treiber stiften. Momentan sind folgende Treiber enthalten:

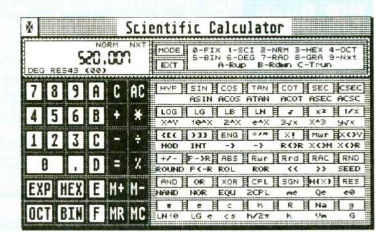
ASCII	CP_80	Epson_12
Epson_LX80	Epson RX80	Gabi 9009
IBM	IBM-Komp	KX-P1092G
LQ_800	MP_165	NEC_P2&P3
NEC_P6&P7	NL10_IBM	NL10_Para
OKI_M182	OKI_M192	QUME
SMM804	TA-SE320	TAXAN 810ST
C 8028	Corona D300	Gabi 9009

16. Diskette
Sammlungen diverser Programme incl. Source Code [s/w] ● *Etikettendruck:* bedruckt Etiketten (Basic) ● *Funkplot:* mit Funktionseingabe (Basic) ● *Kassetten- und Videodatei* unter GEM (Basic) ● *Typewriter:* Schreibmaschinenkurs (Basic) ● *LIFE:* Fortpflanzungsrechnung mehrerer Generationen (Pascal) ● *Schiffe:* Schiffe versenken (Demo für GEM-Routinen) (Basic)

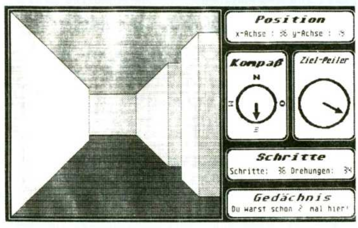
17. Diskette
Programmsammlungsdiskette incl. Source [s/w]
● *Aktien:* Spekulation auf dem Aktienmarkt (Basic) ● *Habu:* Verwaltung von Konten (Basic)
● *Lohnbuchhaltung & Versanddruck:* (Basic)
● *Steuer:* Einkommenssteuer und Lohnsteuerjahresausgleich (Basic)

18. Diskette
● *Doodle plus:* erweiterte Doodle Version; incl. Bilder ● *Power:* Programm zum Einrichten neuer Zeichensätze incl. Sonderzeichensatz [s/w]
● *Konverter:* Wandelt alle Bildformate ● *Datum:* Sichern aktuelles Datum auf Disk ● *Smooth:* Programm zum Scrollen eines 64 000 Byte Doodle-Bildes ● *Format:* Formatierprogramm für diverse Formate ● *Kbug:* auf dem Bildschirm laufen Käfer spazieren, immer und überall.

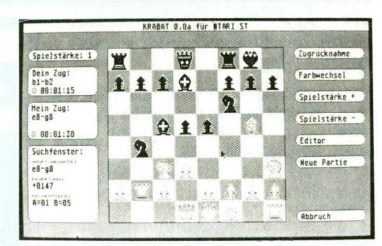
19. Diskette
● *Kermit^plus:* Eine stark verbesserte GEM-Version des Terminalprogramms des Entwicklungspaketes ● *Disksort:* Ein fantastisches Programm zum Ausdruck des Disketteninhaltes entweder als Liste, oder komprimiert für Disketikette



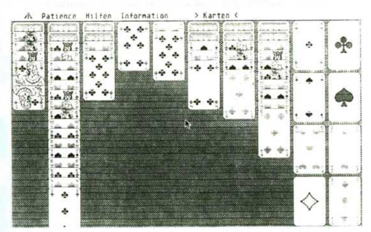
Taschenrechner (PD 12)



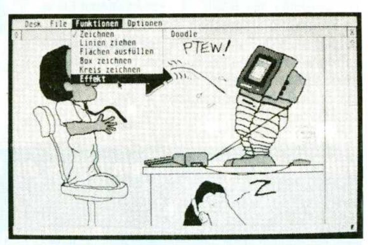
3-D Labyrinth (PD 31)



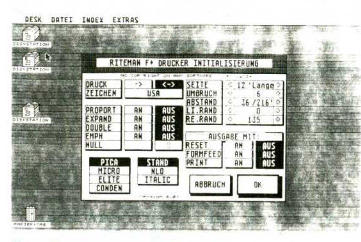
Krabat Schach (PD 10)



Patience (PD 31)



Doodle plus (PD 18)



Druckersteuerung (PD 29)

● *Terminal:* Ebenfalls ein leistungsstarkes Terminalprogramm ● *Fontedit:* Ein Zeichensatzeditor für GEM-Draw und Easy-Draw Fonts [s/w] ● *OKI-Init.ACC:* schickt Steuercodes zu einem Microline 182

20. Diskette
● *Harddisk:* Treiber für Harddisk, der automatisch von Platte geladen wird (keine Bootdisdkette nötig) ● *Toram:* Kopiert bestimmte Programme in eine Ramdisk ● *RSCAnalyse:* Dient zur Analyse von RCS-Files [s/w] ● *DESK Uhr.ACC:* Zeigt ständig Zeit und Datum an. Drei einstellbare Weckzeiten. Läuft mit der ST-Computer Uhr ● *GEMTOOL:* Rahmenprogramm für GEM-Programmierung [s/w]

21. Diskette
● *Adress:* Hilfreiches Programm zur privaten Adressverwaltung ● *Etikett:* Druckt Etiketten aller Art ● *Haushalt:* Verwaltung von Privatkonten; grafische Auswertung ● *Selektor:* Dateiverwaltung für große Datenmengen (max. 30 frei erstellbare Felder)

22. Diskette
● *Plotfit:* grafische Darstellung von Wertepaaren ● *Myterm:* Terminalprogramm unter GfA-Basic ● *Vokabelpauker:* luxuriöser Vokabeltrainer mit vielen Besonderheiten

23. Diskette
● *DGDB:* 'Das große deutsche Ballerspiel', Aktionspiel mit vielen Szenen, Labyrinthen und äußerst beweglichen Figuren [f] ● *Kalah:* Das bekannte Brettspiel ● *Fractals:* Eine schnelle 'Apfelmännchen-Version' in Assembler.

24. Diskette
● *Taurus:* Strategiespiel für mehrere Spieler. Ihre Aufgabe ist es, Städte zu verwalten, Ihren Besitz zu mehren und ihn gegen Gegner zu verteidigen. Sehr schöne Grafik, viele Optionen [f], (leider nur für 1-Mega-Rechner)

25. Diskette
● *Disk Engineer:* Diskutility (Rescratch, ...) ● *Omikron Basic:* Run-Only Interpreter des neuen Basics

26. Diskette
● *Codetabelle:* Gibt jederzeit eine ASCII-Tabelle auf den Bildschirm (Accessoir) [s/w] ● *Life_Game:* Eine Accessoir-Version des bekannten Generationspiels [s/w] ● *Term:* Und noch ein Terminalprogramm mit vielen Fähigkeiten ● *Biorhythmus:* ST-Basic schlägt zurück [s/w] ● *Konto:* Kontoführung in GfA-Basic [s/w] ● *KX P1091:* Accessoir zur Einstellung dieser oder ähnlicher Drucker

27. Diskette
● *NEC-Treiber:* Aufatmen! Dieser Treiber erzeugt in vielen Formaten, gestochen scharfe Hardcopies auf P5/P6/P7 über ALT+HELP ● *ST-Sound Demo:* Die Siegerdemo des ST-Computer Sound Wettbewerbs. 'Zottl, Knurp und Schnatter hauen rein' (siehe Titelbild ST-Computer Nr. 12)

28. Diskette
● *GfA Basic Programme incl. Source* ● *Funkplot:* Es gab schon viele, dieser ist besser [s/w] ● *Terminal:* Ein Terminalprogramm in GfA [s/w] ● *Konverter:* Konvertiert ST-Basic in GfA-Basic ● *ST-Kill:* Utilityprogramme ● *Sprite-Editor:* Sprites Editieren, Verschieben, Speichern, ... [s/w]

29. Diskette
● *Diskutility:* Der Weg ans Directory (manipuliert Löschschutz, erzeugt waschechtes IBM-Format, und mehr) [incl. Source in Pascal] ● *Adress_pas:* Adressverwaltung, die einigen käuflichen Programmen das Fürchten lehrt. ● *Riteman:* Accessoir zum Einstellen eines solchen oder ähnlichen Druckers [incl. Source in Pascal] ● *ST-Graph:* GfA-Basic-Version des Grafikprogrammes aus ST Nr. 1 [s/w] [incl. Source in Basic]

30. Diskette: Spielesammlung
● *Bohnen* ● *Schiffe:* Schiffe versenken ● *Mastermind:* Superhirn für Superhirne [f] ● *Dragon_hunt:* Ein äußerst gefräßiger Drachen will ausgerechnet Ihnen an den Kragen (schönes Spiel) [f] ● *Memo:* Memory (2 starke Versionen) [s/w] ● *Solitaire:* Das bekannte 'Taschenspiel' für den ST [s/w] ● *4 Gewinnt:* Nochmal, aber diesmal in GfA-Basic mit Source [s/w]

31. Diskette: Spielesammlung
● *Patience:* Wunderschöne Version dieses Kartenspiels (siehe Spieletest Nr. 2/87 [s/w] ● *3D-Irrgarten:* Finden Sie einen Weg aus einem dreidimensionalen Labyrinth. Trotz ST-Basic schnell und gut [s/w] ● *Avalun:* Sie verwalten ein Königreich. Kaufen, verkaufen und das Volk versorgen, sonst gibt's Revolution (ST-Basic) ● *YAGL-Schach:* Schach-Programm [s/w]

32. Diskette
● *Mathlib:* Eine umfangreiche Mathematikbibliothek für Pascal. Trigonometrische, Matrizenalgebra und vieles mehr, direkt zum Einbinden ● *Finanzmathe:* Berechnung von Zinsen, Laufzeiten, Lufzeit, Tilgungsplänen, etc.

33. Diskette: diverse Utilities
● *Encrypt:* Programm zum Ver- und Entschlüsseln von Dateien ● *Ramdisk:* Ramdisk beliebiger Größe ● *Recover:* Regeneriert gelöschte Files (!!) ● *Sterm:* VT52 Terminal + Source in Assembler ● *Freeram:* Zeigt freien Speicher

34. Diskette: Der Assembler zum Kurs
● *Assemblerpaket nach M 68000-Standard* incl. 2-Pass Assembler, Disassembler, Debugger, Editor. Direktassemblierung, Labelverarbeitung.

35. Diskette
● *Andromeda:* Zeichenprogramm (DEGAS-ähnlich) ZOOM, Beschriften in vier Richtungen und allen Größen, Kopieren, Linienstil / Art / Breite und alle normalen Grafikbefehle [s/w] ● *3D-Konstruktion:* Zum Entwurf dreidimensionaler Gegenstände, welche aus allen Richtungen abgebildet werden [s/w] ● *Globus:* Animierte Erdballgrafik, zeigt verschiedene Bewegungsabläufe des blauen Planeten

Weiterhin können Sie die letzte TOS-Version (196 480 Bytes) vom 6.2.1986 bei uns, gegen einen Unkostenbeitrag von 10,– DM, erhalten. Zum fehlerfreien Betrieb des Rechners benötigen Sie diese Version, da z. B. mit älteren Versionen manche Ordner nicht gelesen werden können (0 Bytes free). Seltsamerweise liefert ATARI bei neuen Rechnern immer noch das alte TOS aus.
Bis auf die Höhe des Unkostenbeitrags gelten die gleichen Versandbedingungen wie bei der Public Domain Software.
[f] = nur Farbe [s/w] = nur monochrom

Mit 5 Mark sind auch Sie dabei!

Versandbedingungen:

Für eine problemlose und schnelle Bearbeitung müssen Sie folgendes beachten:

- Schicken Sie uns nur **einseitig formatierte** Disketten (3,5 Zoll), auf denen die Diskettennummer und **ihre Adresse** vermerkt sind.
- Außerdem einen ausreichend frankierten Rückumschlag (DM 0,80 reichen nicht), auf dem Ihre Adresse steht. Ferner sollte unsere Adresse als Absender und die Nummern der gewünschten Disketten vermerkt sein.
- In Ihrem eigenen Interesse verwenden Sie bitte gepolsterte Umschläge mit der Aufschrift „Datenträger".
- Außerdem vermerken Sie bitte auf dem uns zugesandten Brief „Public Domain".
- Legen Sie pro Diskette DM 5,– als Verrechnungsscheck bei. (Bei TOS DM 10,–).
- Zu dem Kostenbeitrag müssen wir leider pro Scheck eine Scheckgebühr von DM 0,50 berechnen. Verwenden Sie deshalb nur einen Scheck, auf dem die Gesamtsumme steht (z. B. für 2 Disketten = DM 10,50).
- Bei Versendung der Disketten als **Paket** erhöhen Sie den Betrag bitte um 3,– DM, da dieser Betrag bei Anlieferung zusätzlich entsteht.
- Für ausländische Besteller sei erwähnt, daß Sie mit Euroschecks auch in DM bezahlen können.
- Bitte stellen Sie die Sendungen Zollfrei zu, da uns sonst zusätzliche Kosten entstehen.
- Dieser Kopierservice ist sehr aufwendig; darum bitten wir Sie, **keine** zusätzlichen Bestellungen (Abo, Zeitschrift, Platinen etc.) einzuschließen. Benutzen Sie gegebenenfalls die dafür vorgesehenen Bestellkarten.
- Bitte ausschließlich bei folgender Adresse bestellen:

ST-Computer, Redaktion, Postfach 59 69, 6236 Eschborn 1

BUCH- UND SOFTWARE-BESTELLKARTE

Ich/Wir bestelle(n)

Menge	Bestell-Nr.	Autor/Titel	Stückpreis	gesamt DM
		GfA-BASIC-BUCH	**49,–**	
		Programm-Diskette zum GfA-Buch	**39,–**	

☐ Scheck über DM________ ist beigefügt
☐ Per Nachnahme

Zuzüglich DM 5,– Versandkostenanteil.

Bitte beachten:
Es werden nur Festbestellungen berücksichtigt.
Eine Rückgabemöglichkeit besteht nicht.
Ausnahme nur bei Beschädigung.

Name

Vorname

Straße/Nr.

PLZ Ort

Datum

Unterschrift

ST-COMPUTER Kleinanzeigen-Auftrag

Bitte veröffentlichen Sie für mich folgende Kleinanzeige in der angekreuzten Rubrik

Biete an ☐ Hardware ☐ Software
Ich suche ☐ Hardware ☐ Software
☐ **Tausch** ☐ **Kontakte**
☐ Verschiedenes ☐

30 Buchstaben je Standardzeile – incl. Satzzeichen und Wortzwischenräume.
Groß- und Kleinbuchstaben verwenden, fettgedruckte Wörter unterstreichen.

Bearbeitung **nur gegen Vorausscheck** über den entsprechenden Betrag (keine Überweisung)

☐ privat = DM 7,– je Zeile incl. MwSt.
☐ gewerblich = DM 15,– je Zeile + MwSt.
☐ Chiffregebühr = DM 10,–

☐ Scheck über DM________ ist beigefügt

Bei Angeboten: Ich bestätige, daß ich alle Rechte an den angebotenen Sachen besitze.

Datum Unterschrift

Absenderangaben auf der Rückseite nicht vergessen

Kurzmitteilung

an die ST-Computer Redaktion

Zu dem Artikel __________ in Heft____, Seite____ hätte ich folgendes zu bemerken:

☐ Ich möchte Ihnen folgendes Programm zur Veröffentlichung anbieten: (Kurzbeschreibung, Sprache, Länge in Druckerseiten, GEM/TOS)

☐ Ich kann über folgendes Thema berichten: (Tips & Tricks am ST, Hardware, Software, etc.)

☐ Ich möchte gerne Autor in der ST-Computer werden. Meine Fachgebiete: (z. B. LISP, Pearl, Modula-2, DBase, Assembler...)

☐ Ich möchte, daß folgendes Public-Domain Programm in Ihre Sammlung aufgenommen wird.

☐ Sonstiges

Bei weiteren Angaben oder Fragen wenden Sie sich bitte schriftlich oder telefonisch an die Redaktion. Tel. 06196/482158

ST-COMPUTER Abonnement

Ja, bitte senden Sie mir die ST-Computer Fachzeitschrift ab________
für mindestens 1 Jahr (11 Hefte) zum ermäßigten Preis von jährlich DM 60,– frei Haus.
(Ausland: Nur gegen Scheck-Voreinsendung DM 80,– Normalpost.)
Der Bezugszeitraum verlängert sich nur dann um ein Jahr, wenn nicht 6 Wochen vor Ablauf des Abonnements gekündigt wird.

Name

Vorname

Straße/Nr.

PLZ Ort

Gewünschte Zahlungsweise bitte ankreuzen

☐ Bequem und bargeldlos durch Bankeinzug

Konto-Nr. BLZ

Institut Ort

☐ Ein Verrechnungsscheck über DM______ liegt bei.

☐ Gegen Rechnung

Garantie:
Diese Bestellung kann ich schriftlich innerhalb einer Woche (rechtzeitige Absendung genügt) widerrufen. Dies bestätige ich durch meine 2. Unterschrift.

Datum Unterschrift

Datum Unterschrift

Absender

Name

Vorname

Straße/Postfach

PLZ/Ort

Telefon:

Postkarte

Heim-Verlag
Heidelberger Landstraße 194

6100 Darmstadt-Eberstadt

Telefon 0 61 51 / 5 60 57

Absender

Name, Vorname

Straße

PLZ/Ort

Telefon

Postkarte

Heim-Verlag
ST-Computer
Redaktion
Schwalbacherstr. 64

6236 Eschborn

ST - Kleinanzeigen

Name

Vorname

Firma

Straße/Postfach

PLZ Ort

Datum Unterschrift

Bei Jugendlichen unter 18 Jahren bitte Unterschrift des Erziehungsberechtigten.

Postkarte

Bitte mit 60 Pf. frankieren

Anzeigenservice
Heim-VERLAG
Heidelberger Landstraße 194

6100 Darmstadt-Eberstadt

Telefon 0 61 51 / 5 53 75

ST ABO

Straße/Postfach

PLZ Ort

Telefon:

Postkarte

Bitte mit 60 Pf. frankieren

Heim-VERLAG
Heidelberger Landstraße 194

6100 Darmstadt-Eberstadt

Telefon 0 61 51 / 5 53 75

1000 Berlin



u. a. alphatronic, atari, commodore, dai, epson, sord mit pips, nec hard-/software nach maß — servicetechnik

Kurfürstendamm 121a, 1000 Berlin 31 (Halensee)
Telefon 030/891 1082

Computare

Keithstr. 18-20 • 1000 Berlin 30
030/21 390 21
186 346 com d

DATAPLAY

Bundesallee 25 · 1000 Berlin 31
Telefon: 030/861 91 61

Digital-Computer

Knesebeckstr. 76 · 1000 Berlin 12
Telefon
030-8 82 77 91







Vertragshändler

UNION ZEISS

Kurfürstendamm 57 • 1000 Berlin 15
Telefon 32 30 61

1000 Berlin



Büroelektronik
Keithstraße 26 · 1000 Berlin 30
☎ 26 111 26

2000 Hamburg

Gerhard u. Bernd Waller GbR
Computer & Zubehör

Kieler Straße 689
2000 Hamburg 54

☎ **040/570 60 07 + 570 52 75**

Bit Computer Shop
Osterstraße 173 · 2000 Hamburg 20
Telefon: 040/49 44 00

Createam
Computer Hard & Software
Bramfelder Chaussee 300 2000 Hamburg 71
Telefon Sa Nr. 0 40 / 641 50 91

2000 Norderstedt



Ulzburger Str. 2
2000 Norderstedt
Tel. 0 40 / 527 30 47

2120 Lüneburg

Sienknecht
Bürokommunikation
Beratung - Verkauf - Werkstatt

Heiligengeiststr. 20, 2120 Lüneburg
Tel. 0 41 31 / 4 61 22, Btx 40 24 22
Mo.-Fr. 9⁰⁰-18⁰⁰ und Sa. 9⁰⁰-13⁰⁰

2160 Stade



Büromaschinen · EDV-Systeme
Neue Straße 5, 2160 Stade
Telefon: (04141) 23 64 + 23 84

2210 Itzehoe

Der Computerladen
Inhaber Ulrich Bübel Martin Kopplow

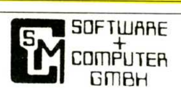
Coriansberg 2 · 2210 Itzehoe
Telefon (0 48 21) 33 90/91

2300 Kiel



Die Welt der Computer
Dreiecksplatz Nr. 7
2300 Kiel 1 · ☎ 04 31 / 56 70 42

2350 Neumünster



Klosterstraße 2 · 2350 Neumünster
Telefon (0 43 21) 4 39 33

2390 Flensburg



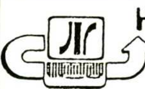
2800 Bremen

PS-DATA

Doventorsteinweg 41
2800 Bremen
Telefon 04 21 - 17 05 77

2850 Bremerhaven

HEIM- UND PERSONALCOMPUTER



Kurt Neumann
Georgstraße 71
2850 Bremerhaven
Tel. 04 71 / 30 21 29

HARDWARE · SOFTWARE · PAPIERWARE

Das ATARI ST-Computer Sonderheft Nr. 1 gibt es jetzt auch bei Ihrem ATARI-Fachhändler.

2900 Oldenburg



COMPUTER-CENTER
Atari · Schneider · Commodore
Multitech · Cumana · Taxan

2900 Oldenburg · Heiligengeiststr. 6
Tel. 04 41 - 40 45 89

2940 Wilhelmshaven

Radio Tiemann GmbH & Co. KG
2940 Wilhelmshaven
Telefon 0 44 21/2 61 45

2950 Leer

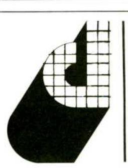


- HARDWARE-SOFTWARE
- SYSTEM-ENTWICKLUNG
- ORGANISATION
- EDV-SCHULUNG
- EDV-BERATUNG
- SERVICE-WARTUNG

Augustenstraße 3 · 2950 Leer
Telefon 04 91 - 45 89

3000 Hannover

Pro-Computer Hannover
Inh. HELGA PROSCHEK
● Beratung ● Verkauf
● Programmierung ● Installation ● Service
☎ 05 11 / 52 25 79
D-3000 Hannover 71 · Großer Hillen 6



DATALOGIC COMPUTERSYSTEME
ATARI ST-COMPUTER · HARDWARE · SOFTWARE
BERATUNG · SERVICE · VERKAUF
CALENBERGER STR. 26
3000 HANNOVER 1
TEL. 0511 - 32 64 89

ATARI ST-Computer Sonderheft Nr. 1:
über 180 Seiten, jetzt bei Ihrem Fachhändler.

COM DATA
Am Schiffgraben 19 · 3000 Hannover 1
Telefon 05 11 - 32 67 36



IBM · EPSON · TRIUMPH ADLER
HEWLETT PACKARD · ATARI etc.
Computer
trendDATA Computer GmbH
Am Marstall 18-22 · 3000 Hannover 1
Telefon (05 11) 1 66 05-0

3040 Soltau

F & T Computervertrieb
Am Hornberg 1
(Industriegeb. Almhöhe)
3040 Soltau
Tel. 0 51 91 / 1 65 22

3100 Celle

Ludwig Haupt jr.
Büro-Einkaufs-Zentrum
Gerhard-Kamm-Straße 2
Ruf 8 30 45, Postfach 140
3100 Celle

3170 Gifhorn

COMPUTER-HAUS GIFHORN
Braunschweigerstr. 50
3170 Gifhorn
Telefon 0 53 71 - 5 44 98

3300 Braunschweig

COMPUTER STUDIO BRAUNSCHWEIG
Rebenring 49-50
3300 Braunschweig
Tel. (05 31) 33 32 77/78

3400 Göttingen

Büroeinrichtungs-Zentrum
Wiederholdt
3400 Göttingen-Weende
Wagenstieg 14 – Tel. 05 51/3 40 31

3500 Kassel

Hermann Fischer GmbH
autorisierter ATARI-Fachhändler
Rudolf-Schwander-Str. 5, 9 + 13
3500 Kassel
Tel. (05 61) 70 00 00

3550 Marburg

L W M
COMPUTER SERVICE
Bahnhofstraße 26 b
3550 Marburg/Lahn
✆ 0 64 21 - 6 22 36

4000 Düsseldorf

BERNSHAUS GmbH
Bürotechnik – Bürobedarf
Cäcilienstraße 2
4000 Düsseldorf 13 (Benrath)
Telefon 02 11 - 71 91 81

4000 Düsseldorf

HOCO
EDV ANLAGEN GMBH
Flügelstr. 47
4000 Düsseldorf
Tel. 02 11 - 77 62 70

4050 Mönchengladbach



Hindenburgstr. 249
4050 Mönchengladbach
Tel. 0 21 61 - 1 87 64

4200 Oberhausen

KAMP
Büro- und Computersysteme
Vestische Straße 89/91
4200 Oberhausen 12 (Osterfeld)
Fernruf (02 08) 89 00 86
Fernschreiber 8 56 578

4290 Bocholt

Systemlösungen
A. W.-Karlen
Alfred-Flender-Str. 284 · 4290 Bocholt
Tel. 0 28 71 - 18 34 89
OKIDATA-Fachhändler

4300 Essen

Computerservice
Keese
Haus-Berge-Straße 163 · 4300 Essen 11
☎ (02 01) 66 70 62

ATARI Systemfachhändler



KARSTADT Aktiengesellschaft
Limbecker Platz 4300 Essen 1
Tel.: (02 01) 17 63 99

4400 Münster



COMPUTER SYSTEME GMBH
Daimlerweg 39 - 4400 Münster
Telefon 02 51 / 71 99 75 - 9

4422 Ahaus

ATARI · Epson · Fujitsu Molecular · NCR · Tandon · Schneider · Star

OCB

OCB-Computershop
Wallstraße 3
4422 Ahaus
Tel. 0 25 61/50 21

OCB-Hard- und Software
Wessumerstraße 49
4422 Ahaus
Tel. 0 25 61/50 21

4430 Steinfurt

ATARI SCHNEIDER STAR EPSON

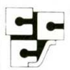
Computer
Büromaschinen
Service
4430 Steinfurt · Tecklenburger Str. 27
Telefon: 0 25 51 / 25 55

4500 Osnabrück

Heinicke-Electronic
Kommenderiestr. 120·4500 Osnabrück
Telefon 05 41 - 8 27 99
Wir liefern Micro-Computer seit 1978

4600 Dortmund

Bürostudio BOLZ
Brauhausstraße 4 · 4600 Dortmund
Telefon 02 31 - 52 77 13 - 16



Atari, Genie, Schneider, Tandy, Brother, Star, Memorex, BASF, Verbatim
cc Computer Studio GmbH
Software-Hardware-Beratung
Service-Eilversand
Ihre Ansprechpartner:
v. Schablinski
Jan P. Schneider T. 0231/528184 · Tx 822631 cccsd
Elisabethstraße 5
4600 Dortmund 1

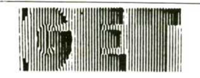
City Elektronic
Güntherstraße 75
4600 Dortmund
Telefon 02 31/57 22 84



4700 Hamm



4790 Paderborn



GESELLSCHAFT FÜR ELEKTRONISCHE TELEKOMMUNIKATION
IM SCHILDERN 15
4790 PADERBORN
TEL. (0 52 51) 2 60 41
BTX ★51051#

4800 Bielefeld

hardware
software
organisation
service
CSF
CSF COMPUTER & SOFTWARE GMBH
Heeper Straße 106 – 108
4800 Bielefeld 1
Tel. (05 21) 6 16 63



Carl-Severing-Str. 190
4800 Bielefeld 14
Telefon: 05 21 / 45 99-150
Telex : 9 37 340 krab d
Telefax: 05 21 / 45 99-123

MICROTEC
Software
Hardware
Beratung
Service

ST-Computer Einkaufsführer:
Werbewirksam, aktuell und preiswert

4930 Detmold

Frevert-Computer
Autorisierter Atari System-Händler
4930 Detmold
Bismarckstr. 12 **Tel. 0 52 31 - 2 24 16**
4920 Lemgo
Braker Mitte 9 **Tel. 0 52 61 - 8 85 20**
8 84 40
Autorisierter Commodore-Systemhändler für Lippe, Minden-Lübecke u. Herford

5000 Köln



5000 KÖLN 1
RICHARD-WAGNER-STR. 39
RUF: 02 21 / 21 91 71

5010 Bergheim

Computerstudio HÖLSCHER
EDV-Beratung · Organisation
Programmierung · Home/Personal-Computer
Software · Zubehör · Fachliteratur
Zeppelinstr. 7 · 5010 Bergheim
Telefon 0 22 71 - 6 20 96

5060 Bergisch-Gladbach

Computer Center
Buchholzstraße 1
5060 Bergisch-Gladbach
Telefon 0 22 02 - 3 50 53

5090 Leverkusen

Rolf Rocke
Computer-Fachgeschäft
Auestraße 1
5090 Leverkusen 3
Telefon 0 21 71 / 26 24

5200 Siebgurg

Computer Center
Luisenstraße 26
5200 Siegburg
Telefon 0 22 41/6 68 54

5222 Morsbach

Computersysteme von A – Z
multicomp
M.-Wallerhausen · Talstr. 22
Tel.: 0 22 94 / 73 08

5400 Koblenz

SCHMITT COMPUTERSYSTEME
Casinostraße 40
5400 Koblenz
✆ 02 61 - 3 65 28

5412 Ransbach

Computer Technik Kieckbusch GmbH
Der Softwarespezialist
Am Seeufer 11 + 22 · 5412 Ransbach
Telefon 0 26 23 - 16 18

5457 Straßenhaus

DR. AUMANN GMBH
Computer-Systeme
Schulstr. 12
5457 Straßenhaus
Telefon 0 26 34 - 40 81/2

5500 Trier



Güterstr. 82 - 5500 Trier
☎ 06 51 - 2 50 44
Fordern Sie unsere Zubehör-Liste an.

5540 Prüm

ATC COMPUTER
J. ZABELL
Kalvarienbergstr. 34
5540 PRÜM
– Tel.: 0 65 51 - 34 83 –

5600 Wuppertal

Jung am Wall
Wall 31—33
5600 Wuppertal 1
Telefon 02 02/45 03 30

5630 Remscheid

C O M SOFT
Scheiderstr. 12 · 5630 Remscheid
Telefon (0 21 91) 2 10 33 - 34

5800 Hagen



Vertragshändler **Axel Böckem**
Computer + Textsysteme
Eiper Str. 60 (Eilpezentrum) · 5800 Hagen
Tel. 0 23 31 / 7 34 90

5900 Siegen



Siegen · Weidenauer Str. 72 · ☎ 02 71/7 34 95

6000 Frankfurt

Müller & Nemecek
Kaiserstraße 44
6000 Frankfurt/M.
Tel. 0 69 - 23 25 44

WAIZENEGGER
Büroeinrichtungen
Kaiserstraße 41
6000 Frankfurt/M.
☎ 0 69 / 23 92 31

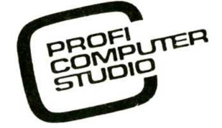
ATARI
... wir machen Spitzentechnologie preiswert.
Vertragshändler
Jetzt bei uns.
bco
Büro-Computer + Organisations GmbH
Oederweg 7—9
6000 Frankfurt/M. 1
☎ (0 69) 55 04 56-57
Hardware ★ Software ★ Beratung ★ Service

SCHMITT
COMPUTERSYSTEME
Grosse Friedbergerstr. 30
6000 Frankfurt
✆ 0 69 - 28 40 65

6100 Darmstadt

Heim
Büro- und Computermarkt
Heidelberger Landstraße 194
6100 Darmstadt-Eberstadt
✆ 0 61 51 / 5 60 57

ATARI Systemfachhändler



KARSTADT Aktiengesellschaft
Elisabethenstr. 15 · 6100 Darmstadt
Luisencenter · Tel. 0 61 51 - 10 94 20

SCHMITT
COMPUTERSYSTEME
Mühlstraße 76
6100 Darmstadt
✆ 0 61 51 - 2 45 74

6200 Wiesbaden

SCHMITT
COMPUTERSYSTEME
Rheinstraße 41
6200 Wiesbaden
✆ 0 61 21 - 30 73 30

6240 Königstein

KFC
COMPUTERSYSTEME
Wiesenstraße 18
6240 Königstein
Tel. 0 61 74 - 30 33
Mail-Box 0 61 74 - 53 55

6300 Gießen



Schneider ATARI Commodore
BAUMS
BÜRO · ORGANISATION
Bahnhofstr. 26 · 6300 Gießen
Telefon: 06 41 / 7 10 96

6330 Wetzlar



Fachmarkt
für
Computer u.
Unterhaltungs-
electronic in Wetzlar,
Einkaufszentrum Bahnhofstraße, Tel. (0 64 41) 4 85 66

6400 Fulda

Schneider ATARI Commodore
WEINRICH
BÜRO · ORGANISATION
Ronsbachstraße 32 · 6400 Fulda
Telefon: 06 61 / 4 92 - 0

6457 Maintal

Landolt-Computer
Beratung · Service · Verkauf · Leasing
Wingertstr. 112
6457 Maintal/Dörnigheim
Telefon 0 61 81 - 4 52 93

6500 Mainz

:ELPHOTEC
Computer Systeme
Ihr Atari Systemhändler mit eigenem Service-Center
Schießgartenstraße 7
6500 Mainz
Telefon 0 61 31 - 23 19 47

SCHMITT COMPUTERSYSTEME
Karmeliterplatz 4
6500 Mainz
✆ 0 61 31 - 23 42 23

6520 Worms

ORION
Computersysteme GmbH
Friedrichstraße 22
6520 WORMS
Tel. 0 62 41 / 67 57 - 67 58

6700 Ludwigshafen

MKV Computermarkt
Bismarck-Zentrum
6700 Ludwigshafen
Telefon 06 21 - 52 55 96

6720 Speyer

MKV Computermarkt
Gilgenstraße 4
6720 Speyer
Telefon 06 232 - 7 72 16

6730 Neustadt

Felten & Meier
Computersysteme
Atari + OKIDATA Fachhändler
Exterstr. 4 · 6730 Neustadt
Tel. 0 63 21 / 8 89 94

6750 Kaiserslautern

C.O.S.
COMPUTER ORG. GmbH
Karl-Marx-Straße 8
6750 Kaiserslautern
Telefon (06 31) 6 50 61 - 62

6750 Kaiserslautern



6800 Mannheim



Computersysteme + Textsysteme
6800 Mannheim 24
Casterfeldstraße 74-76
☎ (06 21) 85 00 40 · Teletex 6 211 912

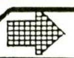
Computer-Center
am Hauptbahnhof GmbH
L 14, 16-17
6800 Mannheim 1
Tel. (06 21) 2 09 83 / 84

6900 Heidelberg

JACOM COMPUTERWELT
Hardware · Software
Schulung · Service
Mönchhofstraße 3 · 6900 Heidelberg
Telefon 0 62 21 / 41 05 14 - 550

Heidelberger Computer-Center
Bahnhofstraße 1
6900 Heidelberg
Telefon 0 62 21 / 2 71 32

7000 Stuttgart


BNT
COMPUTERFACHHANDEL

Der Atari Spezialist hat:
- *über 500 Artikel vorrätig*
- *8 Atari-Fachleute für optimale Kundenbetreuung*
- *Atari-Fachwerkstatt*
- *Soft- und Hardwarelösungen*
- *Schulungen und Clubs*

Fordern Sie unsere Preisliste an!

Telefon (07 11) 55 83 83 / 55 83 91
Telex
BNT Computerfachhandel GmbH
Marktstraße 48 · Stuttgart 50
Mailbox

SCHMITT COMPUTERSYSTEME
Tübingerstr. 18
7000 Stuttgart

7022 L.-Echterdingen

Autorisierter ATARI-System-Fachhändler
ATARI® ST
Matrai computer
Matrai Computer GmbH
Bernhauser Str. 8
7022 L.-Echterdingen
☎ (07 11) 79 70 49

7030 Böblingen

mca
Computer Center
Sindelfinger Allee 1
7030 Böblingen
Tel. 0 70 31/22 36 18

7070 Schwäb. Gmünd



Schwerzerallee 23
7070 Schwäbisch Gmünd
Telefon 0 71 71 / 56 33

7100 Heilbronn

Unser Wissen ist Ihr Vorteil
Walliser & Co.
Mönchseestraße 99
7100 Heilbronn
Telefon 07131/60048



Computer-Welt
Am Wollhaus 6
7100 Heilbronn
Tel. 0 71 31 - 6 84 01 - 03

7150 Backnang



7410 Reutlingen

Computer-Shop
Werner Brock
autorisierter Fachhändler f. ATARI, Schneider, Oki
Federnseestr. 17 · 7410 Reutlingen
Telefon: 0 71 21 / 3 42 87

7450 Hechingen

SRE
Gesellschaft für Datenverarbeitung mbh
Computer · Drucker
Zubehör · Fachliteratur
Schloßplatz 3 · 7450 Hechingen
Telefon 07471/14507

7475 Meßstetten

Ihr ATARI-Systemhändler im Zollern-Alb-Kreis
HEIM + PC-COMPUTERMARKT
HARDWARE · SOFTWARE · LITERATUR
SCHEURER
ATARI COMMODORE CUMANA DATA-BECKER
MULTITECH RITEMAN SCHNEIDER THOMSON
7475 Meßstetten 1 · Hauptstraße 10 · 07431/61280

7480 Sigmaringen



7500 Karlsruhe

papierhaus erhardt
Am Ludwigsplatz · 7500 Karlsruhe
Tel. 0721-23925

MKV Computermarkt
Rüppurer Straße 2d
7500 Karlsruhe
Telefon 0721-373071

7530 Pforzheim

DM Computer GmbH
Hard- & Software
Durlacherstr. 39
Tel. 07231-13939
7530 Pforzheim
Telex 783248

8000 München

schulz computer
Schillerstraße 22
8000 München 2
Telefon (089) 597339
Beratung · Verkauf · Kundendienst

7600 Offenburg

FRANK LEONHARDT ELECTRONIC
Ihr Fachgeschäft für Microcomputer · Hifi · Funk
In der Jeuch 3
7600 Offenburg
Telefon 0781/57974

7640 Kehl/Rhein



eigener Service · eigene Software
Badstrasse 12
Tel. 07807/822
Telex: 752913
7607 NEURIED 2
Filiale:
Hauptstrasse 44
Tel. 07851/1822
7640 KEHL/RHEIN
ELEKTRO-MÜNTZER GmbH

7700 Singen

U. MEIER
Computersysteme
7700 Singen-Htwl.
Am Posthalterswäldle 8
Telefon 07731-44211

7730 VS-Schwenningen

BUS BRAUCH & SAUTER COMPUTER TECHNIK
Villinger Straße 85
7730 VS-Schwenningen
Telefon 07720/38071-72

7750 Konstanz

computertechnik
rösler
D-7750 Konstanz
Zasiusstr. 35 · ☎ 07531/21832

7800 Freiburg

CDS EDV-Service GmbH
Windausstraße 2
7800 Freiburg
Tel. 0761-81047

computer aktuell
Südbadens kompetenter Computer-Partner.
Kaiser-Joseph-Str. 232
7800 Freiburg, Tel.: 0761/2180225

7890 Waldshut-Tiengen

hettler-data
service gmbh
Lenzburger Straße 4
7890 Waldshut-Tiengen
Telefon 07751/3094

7900 Ulm

HARD AND SOFT COMPUTER GMBH
Ulms großes Fachgeschäft
für BTX, Heim- u. Personalcomputer
Herrenkellergasse 16 · 7900 Ulm/Donau
Telefon 0731/62699

//COMPUTERSTUDIO
Büro & Datentechnik · 2x in Ulm
Claus Wecker
Hafenbad 18/1 + Frauenstr. 28
7900 Ulm/Do.
Telefon (0731) 28076

7950 Biberach

HARD AND SOFT COMPUTER GMBH
Biberachs großes Fachgeschäft
für BTX, Heim- u. Personalcomputer
Schulstraße 6 · Bei der VHF
7950 Biberach · Tel. 07351/12221

8000 München



COMPUTER + BÜROTECHNIK
COMPUTER · SOFTWARE · PERIPHERIE
BERATUNG · TECHN. KUNDENDIENST
INGOLSTÄDTER STR. 62L
EURO-INDUSTRIE-PARK · 8000 MÜNCHEN 45
TELEFON 089/3113066 · TELETEX 898341

SCHMITT COMPUTERSYSTEME
Arnulfstraße
8000 München

comp +carry
COMPAQ
apple computer


Das Computer-Fachgeschäft im HERTIE Kaufhaus Hauptbahnhof und Schwabing
Telefon 595277/348054
Ingolstädter Straße 20-24 · 8 München 45 · 089/3597091
Ein Unternehmen der procomp-Gruppe

ATARI ST-Computer Sonderheft Nr. 1:

über 180 Seiten, jetzt bei Ihrem Fachhändler.

ST-Computer Einkaufsführer:

Werbewirksam, aktuell und preiswert

8670 Hof

COMPUTER-CENTER-BURGER
Spezialist für Personal- und Home-Computer, Programme
Zubehör, Beratung, Service
8670 Hof · Leimitzer Straße 11 · Telefon 09281/40075 Abt. Computer



8700 Würzburg

HALLER GMBH
Fachgeschäft für
Mikrocomputer
Büttnerstraße 29
8700 Würzburg
Tel. 09 31/1 67 05

SCHOLL
BÜROTEAM
Hardware · Software
Service · Schulung
computer center
am Dominikanerplatz
Ruf (09 31) 5 04 88

8720 Schweinfurt

Uhlenhuth GmbH
Computer + Unterhaltungselektronik
Albrecht-Dürer-Platz 2
8720 Schweinfurt
Telefon 0 97 21 / 65 21 54

8753 Mömbris

COMPUTIQUE
DIE EXPERTEN FUR MICROCOMPTER
Im Kahltal Zentrum, 8752 Mombris
Telefon: (060 29) 65 20 oder 14 10
ATARI 520 ST
APRICOT
IBM Komp
Festplatten
Hardware
Software
Schulung

8900 Augsburg

Adolf & Schmoll
Computer
Schwalbenstr. 1 · 8900 Augsburg
Tel. (08 21) 52 85 33 oder 52 80 87
Wir sind außerdem autorisierte
Service-Fachwerkstatt für:
Schneider
commodore
ATARI

8900 Augsburg

ATARI Systemfachhändler



KARSTADT Aktiengesellschaft
Bürgermeister-Fischer-Str. 6-10
8900 Augsburg · Tel. (08 21) 31 53 - 416

SCHMITT
COMPUTERSYSTEME
Frauentorstr. 22
8900 Augsburg
✆ 08 21 - 15 42 68

Alle 12 bisher erschienenen Ausgaben gibt's natürlich bei Ihrem Fachhändler.

8940 Memmingen

EDV-Organisation
Hard- + Software
Manfred Schweizer KG
Benninger Str. 34, Tel. 0 83 31 / 1 22 20
8940 Memmingen

Österreich

A-1020 Wien

W + H.
Computerhandel Ges m.b.H.
Förstergasse 6/3/2 · 1020 Wien
Tel. 02 22 - 35 09 68

A-1040 Wien

Ihr ST-Fachhändler in Wien
Computer-Studio
Wehsner Gesellschaft m.b.H.
1040 Wien - Paniglgasse 18-20
Telefon 02 22 - 65 78 08, 65 88 93

A-8010 Graz



Schweiz

CH-1700 Fribourg

Softy Hard's Computershop
Die ATARI ST Spezialisten
Grand Rue 42
CH-1700 Fribourg
Tel. 0041(0) 37 22 26 28

CH-2503 Biel

UE
URWA Electronic
Ihr ATARI ST Spezialist in der Schweiz.
0 32 / 25 45 53
Lindenweg 24, 2503 Biel

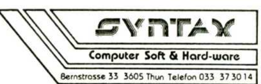
CH-3000 Bern

Computer Corner
S. Bazan
Könizstrasse 70
3008 Bern
✆ 031/25 57 37
Hardware
Software
Zubehör
Occasion

CH-3415 Hasle Rüegsau

COMPU-TRADE
Ihr ATARI Spezialist
Emmenstr. 16
CH-3415 Hasle-Rüegsau
✆ 0 34 / 61 45 93 auch abends bis 21.00 h
HARD- u. SOFTWARE · BERATUNG · EILVERSAND

CH-3605 Thun



Autorisierter ATARI-
System-Fachhändler
für 520 ST

CH-4054 Basel

COMPUTER CENTER
COMPUTER SYSTEME
COMPUTER SCHULE

DIE ST-SPEZIALISTEN IN BASEL

Öffnungszeiten:
Di. – Fr. 9.30 – 12.30 / 14.00 – 18.30
Sa. 9.30 – 16.00
Mo. geschlossen

SYSAG

HOLEESTRASSE 87 · 4054 BASEL · TELEFON 061 39 25 25

CH-4625 Oberbuchsiten

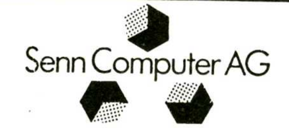
STECTRONIC M. Steck
Electronic-Computer-Shop

Hauptstr. 104/137
CH-4625 OBERBUCHSITEN
Tel. 0 62/63 17 27 + 63 10 27

CH-5400 Baden

A-Magic Computers
Obere Halde 27
CH-5400 Baden

CH-5430 Wettingen



Zentralstrasse 93
CH-5430 Wettingen
Tel. 056 / 27 16 60
Telex 814 193 seco

CH-8006 Zürich

ADAG Computershop

Universitätsstr. 25 · 8006 Zürich
Tel. 01 / 47 35 54

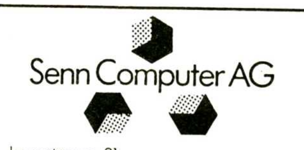
ATARI & WANG & EPSON

CH-8006 Zürich

Computer-Center P. Fisch

Stampfenbachplatz 4
8006 ZÜRICH
✆ 01 / 363 67 67

CH-8021 Zürich



Langstrasse 31
Postfach
CH-8021 Zürich
Tel. 01 / 241 73 73
Telex 814 193 seco



Das Warenhaus der neuen Ideen

Ihr Computer-Fachhandel an der
Bahnhofstrasse 75 · Zürich

CH-8246 Langwiesen

ZIMELEC
Elektronik-Shop
8246 Langwiesen
☎ 0 53 55 22 4

Montag-Freitag von 10-20 Uhr geöffnet
Samstag von 10-16 Uhr.

CH-9000 St. Gallen

VIDEO - COMPUTER - CENTER
GÄCHTER AG



Webergasse 22
9000 St. Gallen
Telefon 071 / 22 60 05

CH-9400 Rorschach

PAUS-electronic
Ihr Computer-Fachhandel
Industriestr. 30

CH-9400 Rorschach ✆ 071-41 18 85

CH-9450 Altstätten

PAUS-electronic
Ihr Computer-Fachhandel

Ringgasse 27 · CH-9450 Altstätten
Telefon 0 71 - 75 34 79

Einkaufsführer

Für ATARI-Fachhändler genau das Richtige.

Werbewirksam, aktuell und preiswert.

Sprechen Sie mit uns:
Heim-Verlag
Tel.: 0 61 51 / 5 60 57

Luxemburg



Feb. '87 — Anzeigenschluß am 6. Januar 1987

März '87 — Anzeigenschluß am 3. Februar 1987

Verspätete Einsendungen kommen in die darauffolgende Ausgabe!

ISAM & PRIMA

Teil 3 – die Standardroutinen

Im vorletzten Teil unserer Serie über die Assembler-Macro-Bibliothek „ISAM & PRIMA" werden die Standardroutinen des Modulpakets vorgestellt – zumeist Arithmetik-Routinen.

Die Artithmetik-Routinen von ISAM & PRIMA arbeiten – besonders bezüglich Kommastellen und Vorzeichen – nach den Regeln der Mathematik. Die Routinen für die Grundrechenoperationen lauten „Plus", „Minus", „Durch" und „Mal".

Zur Berechnung werden zwei standardisierte Rechenfelder mit den Bezeichnungen REFE1 und REFE2 verwendet. Sie haben folgendes Format:

```
          DC "0"
REFE1:    DC "0000000000"
VORZ1:    DC "+"
KOMMA1:   DC 0
SIGNI1:   DC 0
          DC "0"
REFE2:    DC "0000000000"
VORZ2:    DC "-"
KOMMA2:   DC 0
SIGNI2:   DC 0
```

Zur Erläuterung: Das Format wird jeweils eine Stelle vor den Feldern REFE1 und REFE2 benötigt. Dies wird von der Routine „NUMERIC" verlangt, die numerische Werte von der Tastatur holt und in Rechenfelder speichert. Das Feld REFE enthält den numerischen ASCII-Wert mit voranstehenden Nullen, VORZ das jeweilige Vorzeichen (+ oder –), KOMMA die Anzahl der Stellen hinter dem Komma und SIGNI die Position der ersten signifikanten Ziffern relativ zu dem Feld REFE.

Bei Bedarf kann man auch selbst andere Rechenfelder definieren. Das ist etwa beim Zwischenspeichern oder beim Umwandeln von ASCII in Binärformat interessant. Der eigentliche Rechenvorgang findet jedoch stets in den Feldern REFE1 und REFE2 statt.

Die Arithmetik-Routinen werden mit einem JSR (Jump Subroutine) ange-

```
INPUT:     MOVEM.L   D1-D2/A2,-(SP)
           MOVE.B    D0,D1
           EXT.W     D1                  ;MAXIMALE EINGABE -> D1
           CLR.L     D2                  ;ZAHLER TATSACHL.EING.
IN010:     SUBQ.W    #1,D1
           TST.W     D1
           BMI       IN040               ;MAXIMALZAHL ERREICHT
IN020:     INKEY
           TST.L     D0
           BEQ       IN020
           CMP.B     #8,D0               ;BACKSPACE?
           BEQ       IN030               ;JA
           CMP.B     #13,D0              ;CR ?
           BEQ       IN040
           CMP.B     #3,D0               ;^C ?
           BNE       IN025               ;NEIN
           MOVE.L    #-3,D0              ;ANZEIGER F.^C
           MOVEM.L   (SP)+,D1-D2/A2
           CLR.B     (A2)                ;X'00' AN PUFFERANFANG
           RTS
IN025:     ADDQ.B    #1,D2
           MOVE.B    D0,(A2)+            ;ZEICHEN -> PUFFER
           CHAR      D0
           BRA       IN010
IN030:     TST.B     D2                  ;WAREN SCHON ZEICHEN DA ?
           BEQ       IN020               ;NEIN, BACKSPACE IGNORIEREN
           SUBQ.B    #1,D2
           ADDQ.W    #1,D1
           SUBQ.L    #1,A2
           MOVE.B    #32,(A2)            ;VORANGEGANGENES ZEICHEN AUF BLANK
           CHAR      #8
           CHAR      #32
           CONTROL   CLINKS
           BRA       IN020
IN040:     MOVE.L    D2,D0               ;ANZAHL EINGABEZEICHEN
           CLR.B     (A2)                ;X"00" ANS PUFFERENDE
           MOVEM.L   (SP)+,D1-D2/A2
           RTS
```

```
NUMERIC:   MOVEM.L   D1/A0-A2,-(SP)
           MOVE.B    #10,D0              ;MAX.10 ZEICHEN HOLEN
           JSR       INPUT               ;PUFFERADR STEHT SCHON IN A2
NUM000:    TST.B     D0                  ;CONTROL C ?
           BMI       NUMEXIT
           MOVE.B    D0,-1(A2)           ;GELESENE ANZAHL -> TATL
           MOVE.B    #"+",10(A2)         ;MIT PLUS INITIALISIEREN
           TST.B     D0
           BEQ       NUM135              ;NUR ENTER-TASTE
           CMP.B     #"-",(A2)
           BNE       NUM030
NUM010:    MOVE.B    (A2),D0             ;VORZEICHEN RETTEN
           MOVE.L    A2,A0
           MOVE.L    A2,A1
           ADDQ.L    #1,A1
           MOVE      #9,D1               ;10 STELLEN UM 1 STELLE
NUM020:    MOVE.B    (A1)+,(A0)+         ;NACH LINKS
           DBRA      D1,NUM020
           MOVE.B    D0,10(A2)           ;VORZEICHEN
           BRA       NUM040
NUM030:    CMP.B     #"+",(A2)
           BEQ       NUM010
           BRA       NUM045
NUM040:    SUBQ.B    #1,-1(A2)
NUM045:    MOVE.B    -1(A2),D0           ;NACH DEZIMALPKT SUCHEN
           EXT.W     D0                  ;IN LANGE TATL
           SUBQ      #1,D0
           MOVE.L    A2,A0               ;AB ERG
NUM050:    CMP.B     #".",(A0)+
```

sprungen. Bei einem Überlauf enthält das Datenregister D0 den Wert −1.

Ein ungenaues Ergebnis entsteht, wenn zuviele signifikante Stellen vorhanden sind. In diesem Fall muß eine bestimmte Anzahl der Stellen nach dem Komma entfernt werden, da sonst ein Überlauf auftritt. Dadurch, daß diese Stellen abgetrennt wurden, erhält man logischerweise ein ungenaues Ergebnis. Das Feld „GENAU" (.B) enthält dann eine −1.

Falls keine Überlaufanzeige erfolgte und „GENAU" eine 0 enthält, ist das Ergebnis absolut genau.

Nun zu den einzelnen Routinen:

INPUT

- Eingabe von Zeichen von der Tastatur

Vor dem Aufruf mit JSR INPUT muß in D0 die maximale Anzahl der Zeichen und in A2 die Pufferadresse vorhanden sein. Die tatsächliche Anzahl der Zeichen steht im Anschluß an D0. Die eingegebenen Zeichen findet man im Puffer wieder, wobei sie durch X"00" abgeschlossen werden. Der Puffer muß aus diesem Grund um eine Stelle größer als die längste mögliche Eingabe sein. Wurde ein C eingegeben, gleich an welcher Stelle der Eingabe, so steht beim Rücksprung in D0 eine −3 und an der ersten Pufferstelle X"00". Man sollte also nach der Rückkehr aus der INPUT-Routine unbedingt D0 auf negativ oder −3 abfragen.

NUMERIC

- Eingabe von numerischen Zeichen von der Tastatur

Bei dieser Routine müssen zuvor einige Felder definiert werden:

```
TATL:   DC  0
ERG:    BLK 10,"0" ;VOR ANSPRUNG
VORZ:   DC  "+"    ;ADR "ERG" NACH
KOMMA:  DC  0      ;A2 BRINGEN
SIGNI:  DC  0
```

Vor dem Ansprung mit JSR NUMERIC muß die Adresse ERG in A2 stehen. Die eingegebene Zahl findet man nach der Rückkehr aus dieser Routine in ERG (Länge = zehn Stellen mit führenden Nullen). Die Anzahl der Stellen nach dem Komma steht in KOMMA, die Anzahl der signifikanten Stellen in SIGNI. Es wird eine Prüfung durchgeführt, ob das eingegebene

```
          DBEQ      D0,NUM050
          TST       D0
          BMI       NUM080          ;NICHT GEFUNDEN
          SUBQ.L    #1,A0
          MOVE.L    A2,D0           ;ANZAHL NACHKOMMAST. ERRECHNEN
          SUB.L     A0,D0           ;(ADR.ERG + TATL-ADR.PKT - 1)
          SUBQ.L    #1,D0
          MOVE.B    -1(A2),D1
          EXT.W     D1
          EXT.L     D1
          ADD.L     D1,D0
          MOVE.B    D0,11(A2)
          CMP.B     #0,D0
          BEQ       NUM070          ;KEINE NACHKOMMASTELLEN
          MOVE.L    A0,A1           ;PUNKT ENTFERNEN
          ADDQ.L    #1,A1           ;A0=EMPF.-FLD, A1=SENDEFELD
          SUBQ.L    #1,D0           ;IN LANGE NACHKOMMASTELLEN
NUM060:   MOVE.B    (A1)+,(A0)+
          DBRA      D0,NUM060
NUM070:   SUBQ.B    #1,-1(A2)
          BRA       NUM090
NUM080:   CLR.B     11(A2)
NUM090:   MOVE.B    -1(A2),D0       ;RECHTSBÖNDIG VERSCHIEBEN
          EXT.W     D0

          SUBQ      #1,D0           ;IN LANGE TATL
          BMI       NUM130          ;NUR PUNKT EINGEGEBEN
          MOVE.B    -1(A2),D1       ;A0=SENDEFELD
          EXT.W     D1
          EXT.L     D1
          MOVE.L    D1,A0
          ADD.L     A2,A0
          MOVE.L    A2,A1           ;A1=EMPFANGSFELD
          ADD.L     #10,A1
NUM100:   MOVE.B    -(A0),-(A1)
          DBRA      D0,NUM100
          MOVE.B    -1(A2),D0       ;FÜHRENDE NULLEN EINFÜGEN
          NEG.B     D0              ;LANGE= 10 - TATL
          EXT.W     D0
          ADD.W     #9,D0
          TST.W     D0
          BMI       NUM115
          MOVE.L    A2,A0
NUM110:   MOVE.B    #"0",(A0)+
          DBRA      D0,NUM110
NUM115:   MOVE      #9,D0           ;PRÜFUNG AUF NUMERISCH
          MOVE.L    A2,A0
NUM120:   CMP.B     #"0"-1,(A0)
          BLS       NUM130          ;FEHLER
          CMP.B     #"9",(A0)+
          DBHI      D0,NUM120
          TST       D0
          BMI       NUM140          ;OK, NUMERISCH
NUM130:   MOVE.L    #-1,D0
NUMEXIT:  MOVEM.L   (SP)+,D1/A0-A2
          RTS
NUM135:   MOVE.L    #-2,D0
          BRA       NUMEXIT
NUM140:   CMP.B     #"3",(A2)       ;ZAHL > 3.999.999.999 ?
          BHI       NUM130          ;JA, ZU GROß
          MOVE.W    #9,D0           ;1. SIGNIF.ZIFFER ERMITTELN
          MOVE.L    A2,A0
NUM150:   CMP.B     #"0",(A0)+
          DBNE      D0,NUM150
          SUB.L     A2,A0
          SUBQ.L    #1,A0
          MOVE.W    A0,D0
          MOVE.B    D0,12(A2)
          CLR.L     D0
          BRA       NUMEXIT


ABGLEICH: MOVEM.L   D1-D3/A1/A2,-(SP)
          MOVE.L    #REFE1,A1
          MOVE.L    #REFE2,A2
ABG000:   MOVE.B    11(A1),D1
          CMP.B     11(A2),D1       ;KOMMA1 > KOMMA2 ?
          BGT       ABG020          ;JA
          BLT       ABG010          ;KOMMA2 > KOMMA1
          MOVEM.L   (SP)+,D1-D3/A1/A2
          RTS
ABG010:   EXG       A1,A2
ABG020:   MOVE.B    12(A2),D1
          CMP.B     D0,D1           ;MINIMALWERT SIGNI ERREICHT ?
          BLE       ABG030          ;JA
          MOVE.B    12(A2),D1       ;REFE2 UM 1 STELLE NACH LINKS
```

Zeichen numerisch ist. Erfolgt eine falsche Eingabe oder wird nur die RETURN-Taste gedrückt oder ist die eingegebene Zahl größer 3.999.999.999, wird D0 vor dem Rücksprung auf −1 gesetzt. Bei Eingabe von C steht in D0 anschließend eine −3. Alle Register, außer D0, bleiben unverändert.

ABGLEICH

– Kommastellen von REFE1 und REFE2 aufeinander abgleichen

Diese Routine wird von den Arithmetik-Routinen verwendet und ist im eigentlichen Sinne keine Standardroutine.

Vor dem Ansprung der Routine muß in D0 (.B) der Minimalwert stehen, den SIGNI annehmen darf. Als Ergebnis erhält man gleiche Nachkommastellen in REFE1 und REFE2. Das Feld mit der geringeren Anzahl von Nachkommastellen wird dabei dem anderen angepaßt. Falls der Minimalwert für SIGNI unterschritten wird, werden rechts Nachkommastellen abgeschnitten. Alle Register bleiben bei dieser Routine unverändert.

Man kann diese Routine ebenfalls zum Erweitern der Nachkommastellen auf eine bestimmte Anzahl nutzen. Dazu nimmt man eines der Rechenfelder als sogenanntes „Dummy" und gibt im Feld KOMMA die gewünschte Anzahl von Nachkommastellen an, auf die das andere Rechenfeld erweitert werden soll.

UEPLUS

– Überlaufkontrolle für Addition und Subtraktion

Diese Routine ist genau wie ABGLEICH keine eigentliche Standardroutine, wird aber von den Arithmetik-Routinen benutzt. Ein Überlauf in den Rechenfeldern wird durch eine −1 im Register D0 angezeigt. Man sollte die Überlaufkontroll-Routine mit JSR UEPLUS anspringen, nachdem ein Abgleich der Kommastellen stattgefunden hat. UEPLUS versucht dann einen drohenen Überlauf durch Abschneiden von Nachkommastellen abzuwenden.

PLUS

– Addition zweier Werte

Durch Aufruf dieser Routine werden die beiden Rechenfelder REFE1 und

```
          EXT.W     D1
          EXT.L     D1              ;SIGNI2 + ADR REFE2
          ADD.L     A2,D1           ;= ADR SENDEFELD -> D1
          MOVE.L    D1,D2           ;ADR SENDEFELD - 1
          SUBQ.L    #1,D2           ;= ADR EMPFFLD -> D2
          CLR.L     D3              ;LANGE
          MOVE.B    12(A2),D3       ;=
          NEG.L     D3              ;  -SIGNI2
          ADD.L     #10,D3          ;  +10        -> D3
          TRANSFER  D1,D2,D3
          MOVE.B    #"0",9(A2)      ;NULL EINFÜGEN
          ADDQ.B    #1,11(A2)       ;KOMMA2 + 1
          SUBQ.B    #1,12(A2)       ;SIGNI2 - 1
          BRA       ABG000
ABG030:   MOVE.L    A1,D1           ;REFE1 UM 1 STELLE NACH RECHTS
          MOVE.L    D1,D2
          ADDQ.L    #1,D2
          TRANSFER  D1,D2,#9
          MOVE.B    #"0",(A1)
          SUBQ.B    #1,11(A1)
          ADDQ.B    #1,12(A1)
          MOVE.B    #-1,GENAU
          BRA       ABG000


UEPLUS:   MOVEM.L   A1/A2,-(SP)     ;ÜBERLAUFPRÜFUNG
          MOVE.L    #REFE1,A1       ;ADDITION/SUBTRAKTION
          MOVE.L    #REFE2,A2
UEPLUS1:  CMP.B     #2,12(A1)       ;SIGNI1 < 2 ?
          BLT       UEP020          ;JA
UEP000:   CLR.L     D0
UEP010:   MOVEM.L   (SP)+,A1/A2
          RTS
UEP020:   CMP.B     #2,12(A2)       ;SIGNI2 < 2 ?
          BGE       UEP000          ;NEIN
          CMP.B     #0,11(A1)       ;KOMMA1 > 0 ?
          BGT       UEP030          ;JA
          MOVE.L    #-1,D0          ;ÜBERLAUF
          BRA       UEP010
UEP030:   MOVE.L    #UEPLUS1,-(SP)  ;RÜCKSPRUNGADRESSE
          MOVEM.L   D1-D3/A1/A2,-(SP)
          JMP       ABG030


PLUS:     MOVEM.L   D1-D3/A2,-(SP)
          MOVE.B    #2,D0           ;MINIMALWERT SIGNI
          CLR.B     GENAU
          JSR       ABGLEICH
          JSR       UEPLUS
          TST.L     D0
          BEQ       PLU020          ;KEIN ÜBERLAUF
PLU010:   MOVEM.L   (SP)+,D1-D3/A2
          RTS
PLU020:   MOVE.L    #REFE2,A2
          JSR       ASCBI
          MOVE.L    D1,D2
          MOVE.L    #REFE1,A2
          JSR       ASCBI
          CMP.B     #"+",REFE1+10
          BEQ       PLU030
          CMP.B     #"-",REFE2+10
          BEQ       PLU070
          BRA       PLU040
PLU030:   CMP.B     #"+",REFE2+10
          BEQ       PLU060
PLU040:   CMP.L     D2,D1
          BHI       PLU050
          EXG       D1,D2
          MOVE.B    REFE2+10,D3
          BRA       PLU055
PLU050:   MOVE.B    REFE1+10,D3
PLU055:   SUB.L     D2,D1
          BRA       PLU999
PLU060:   MOVE.B    #"+",D3
PLU065:   ADD.L     D2,D1
          BRA       PLU999
PLU070:   MOVE.B    #"-",D3
          BRA       PLU065
PLU999:   JSR       BINASC
          MOVE.B    D3,REFE1+10
          JSR       NULWEG
          BRA       PLU010
```

REFE2 addiert. Das Ergebnis steht anschließend in REFE1.

MINUS

– Subtraktion zweier Werte

Diese Routine benutzt die PLUS-Routine, nachdem zuvor eine Vorzeichen-Invertierung stattgefunden hat. Es werden wieder die beiden Rechenfelder benutzt. Das Ergebnis steht anschließend in REFE1.

DURCH

– Division zweier Werte

Diese Routine dividiert REFE1 und REFE2. Das Ergebnis steht in REFE1.

MAL

– Multiplikation zweier Werte

Durch diese Routine wird REFE1 mit REFE2 multipliziert. Das Ergebnis steht wieder in REFE1.

KNEIF

– Abschneiden von Kommastellen ohlne Runden

Vor dem Aufruf mit JSR KNEIF muß zuvor die Anzahl der gewünschten Nachkommastellen nach dem Abschneiden an D0 übergeben werden. Ebenso muß der Inhalt des Rechenfeldes REFE1 in A2 stehen. Das Ergebnis findet man danach wieder in REFE1.

ROUND

– Runden der Nachkommastellen

Ebenso wie bei KNEIF müssen vor dem Aufruf der Routine die Nachkommastellen an D0 und REFE1 an A2 übergeben werden. Das Ergebnis steht dann wieder in REFE1.

BINASC

– Umwandlung von Binär- in ASCII-Format

Vor dem Aufruf der Routine muß die Anfangsadresse von REFE an A2 übergeben werden. Die Binärzahl muß als Langwort in D1 stehen, wobei Bit 31 zur Zahl und kein Vorzeichen möglich ist. Zur Umwandlung wird die Routine DIVI benötigt. Nach dem Aufruf mit JSR BINASC findet man die ASCII-Zahl im Feld REFE 10-stellig mit führenden Nullen wieder. Das Vorzeichen (immer "+") steht in

```
MINUS:   EOR.B     #6,REFE2+10       ;VORZEICHEN INVERTIEREN
         JSR       PLUS
         EOR.B     #6,REFE2+10
         RTS

DURCH:   MOVEM.L   D1-D5/A0,-(SP)
         MOVE.L    #1,D0             ;MINIMALWERT SIGNI = 1
         CLR.B     GENAU
         JSR       ABGLEICH
         MOVE.B    REFE1+10,D0       ;VORZEICHEN REFE1
         CMP.B     REFE2+10,D0       ;= VORZEICHEN REFE2 ?
         BNE       DU002             ;NEIN, ERGEBNIS NEGATIV
         MOVE.B    #"+",-(SP)        ;JA, ERGEBNIS POSITIV
         BRA       DU004
DU002:   MOVE.B    #"-",-(SP)
DU004:   MOVE.L    #REFE2,A2
         JSR       ASCBI
         MOVE.L    D1,D3             ;REFE2 -> D3
         MOVE.L    #REFE1,A2         ;REFE1 = DIVIDEND
         JSR       ASCBI             ;      -> D1
         TST.L     D3                ;DIVISOR = 0
         BNE       DU006
DU005:   MOVE.B    (SP)+,D0          ;ÜBERLAUF ODER /0, STACKKORREKTUR
         MOVE.L    #-1,D0            ;FEHLERANZEIGER
         BRA       DU999
DU006:   JSR       DIVI
DU007:   JSR       BINASC            ;VORKOMMAST. -> REFE1
         MOVE.L    #REFE1,D4         ;ADREMPF -> D4
         CLR.L     D1                ;SIGNI + ADRREFE1
         MOVE.B    REFE1+12,D1       ;=
         ADD.L     D4,D1             ;ADRSEND -> D1
         MOVE.B    REFE1+12,D0
         NEG.B     D0                ;- SIGNI
         EXT.W     D0                ;+ 10
         ADD.W     #10,D0            ;= LÄNGE -> D0
         TRANSFER  D1,D4,D0
         MOVE.B    REFE1+12,D1
         NEG.B     D1                ;- SIGNI
         EXT.W     D1                ;+ ADRREFE1
         EXT.L     D1                ;+ 10
         ADD.L     #REFE1+10,D1      ;= ADREMPF -> D1
         MOVE.B    REFE1+12,D0       ;LÄNGE = SIGNI -> D0
         EXT.W     D0
         FILL      D1,D0,#"0"        ;RECHTS NULLEN EINFÜGEN
         MOVE.B    REFE1+12,D5
         SUBQ.B    #1,D5
         BMI       DU020
         EXT.W     D5
         EXT.L     D5
         MOVE.L    #0,A0
         SUB.L     D5,A0
         ADD.L     #REFE1+9,A0
DU010:   TST.L     D2                ;REST = 0 ?
         BEQ       DU020             ;JA
         MOVE.L    #10,D1
         JSR       MULTI             ;REST * 10
         TST       D0                ;ÜBERLAUF ?
         BMI       DU005             ;JA
DU015:   JSR       DIVI              ;/ DIVISOR
         OR.B      #$30,D1           ;= NÄCHSTE NACHKOMMASTELLE
         MOVE.B    D1,(A0)+
         DBRA      D5,DU010
DU020:   MOVE.B    REFE1+12,REFE1+11    ;KOMMA = SIGNI
         CLR.B     REFE1+12             ;SIGNI = 0
DUEXIT:  MOVE.B    (SP)+,REFE1+10       ;VORZEICHEN EINSTEUERN
         JSR       NULWEG
         CLR.L     D0
DU999:   MOVEM.L   (SP)+,D1-D5/A0
         RTS
```

```
MAL:      MOVEM.L   D1/D2/A1/A2,-(SP)
          CLR.B     D0
          CLR.B     GENAU
          MOVE.B    REFE1+10,D0
          CMP.B     REFE2+10,D0
          BNE       MA020
          MOVE.B    #"+",-(SP)
          BRA       MA025
MA020:    MOVE.B    #"-",-(SP)
MA025:    MOVE.L    #REFE2,A2
          JSR       ASCBI
          MOVE.L    D1,D2
          MOVE.L    #REFE1,A2
          JSR       ASCBI
          JSR       MULTI
          TST.B     D0
          BPL       MA030
          BRA       MA040
MAEXIT:   MOVE.B    (SP)+,REFE1+10
          JSR       NULWEG
MA999:    MOVEM.L   (SP)+,D1/D2/A1/A2
          RTS
MA030:    MOVE.B    REFE1+11,D0
          ADD.B     REFE2+11,D0
          CMP.B     #10,D0
          BGT       MA045
          MOVE.B    D0,REFE1+11
          JSR       BINASC
          TST.B     D0
          BEQ       MAEXIT
          NEG.B     D0
          ADD.B     #10,D0
          BNE       MA032
          CLR.B     REFE1+12
          BRA       MAEXIT
MA032:    CMP.B     REFE1+12,D0
          BLT       MA034
          BRA       MAEXIT
MA034:    SUBQ.B    #1,D0
          MOVE.B    D0,REFE1+12
          BRA       MAEXIT
MA040:    MOVE.B    REFE1+11,D0
          CMP.B     REFE2+11,D0
          BGT       MA042
          TST.B     REFE2+11
          BEQ       MA045
          MOVE.L    #REFE2,A1
MA041:    JSR       MA060
          MOVE.B    #-1,GENAU
          BRA       MA025
MA042:    MOVE.L    #REFE1,A1
          BRA       MA041
MA045:    MOVE.L    #-1,D0

          ADDQ.L    #2,SP
          BRA       MA999
MA060:    MOVE.L    A1,D1
          MOVE.L    D1,D2
          ADDQ.L    #1,D2
          TRANSFER  D1,D2,#9
          MOVE.B    #"0",(A1)
          SUBQ.B    #1,11(A1)
          ADDQ.B    #1,12(A1)
          RTS
NULWEG:   MOVEM.L   D1/D2/A1,-(SP) ;RECHTSBÜNDIGE NULLEN ENTFERN.
          MOVE.L    #REFE1,A1
NUL000:   CMP.B     #"0",REFE1+9
          BNE       NUL005
          TST.B     REFE1+11
          BNE       NUL010
NUL005:   MOVEM.L   (SP)+,D1/D2/A1
          RTS
NUL010:   MOVE.B    REFE1+12,D1
          CMP.B     #9,D1
          BGE       NUL005
          JSR       MA060
          BRA       NUL000
```

```
ROUND:   MOVEM.L  D1-D3/A3,-(SP)
         MOVE.B   11(A2),D1        ;TATSÄCHL.KOMMAST. -> D1
         SUB.B    D0,D1            ;- GEWÜNSCHTE KOMMAST.
         CMP.B    #1,D1
         BLT      RO020            ;HAT SICH ERLEDIGT
         MOVE.B   D0,11(A2)        ;NEUE = GEWÜNSCHTE KOMMAST.
         EXT.W    D1               ;= VERSCHIEBELANGE -> D1
         EXT.L    D1
         MOVE.L   D1,D2            ;10 - VERSCHIEBELANGE
         NEG.L    D2               ;=
         ADD.L    #10,D2           ;TRANSFERLANGE -> D2
         MOVE.L   D2,A3
         CMP.B    #$34,0(A3,A2)    ;ABKNEIFSTELLE > 4 ?
         BLE      RO010            ;NEIN
         MOVE.B   #-1,D0           ;MERKMAL RUNDEN
RO010:   MOVE.L   D1,D3            ;VERSCHIEBELANGE RETTEN
         ADD.L    A2,D1            ;REFE+VERSCHIEBELANGE=EMPF.-ADR.
         TRANSFER A2,D1,D2
         FILL     A2,D3,#"0"
         CMP.B    #-1,D0           ;RUNDEN ?
         BNE      RO012            ;NEIN
         JSR      ASCBI            ;ZAHL -> D1
         ADDQ.L   #1,D1            ;+1
         JSR      BINASC           ;ZAHL ZURÜCK
RO012:   MOVE.W   #9,D0            ;1.SIGNIF.ZIFF.ERMITTELN
         MOVE.L   A2,A0
RO015:   CMP.B    #"0",(A0)+
         DBNE     D0,RO015
         SUB.L    A2,A0
         SUBQ.L   #1,A0
         MOVE.W   A0,D0
         MOVE.B   D0,12(A2)
RO020:   MOVEM.L  (SP)+,D1-D3/A3
         RTS
```

```
KNEIF:   MOVEM.L  D1-D3/A3,-(SP)
         MOVE.B   11(A2),D1        ;TATSÄCHL.KOMMAST. -> D1
         SUB.B    D0,D1            ;- GEWÜNSCHTE KOMMAST.
         CMP.B    #1,D1
         BLT      KN020            ;HAT SICH ERLEDIGT
         MOVE.B   D0,11(A2)        ;NEUE = GEWÜNSCHTE KOMMAST.
         EXT.W    D1               ;= VERSCHIEBELANGE -> D1
         EXT.L    D1
         MOVE.L   D1,D2            ;10 - VERSCHIEBELANGE
         NEG.L    D2               ;=
         ADD.L    #10,D2           ;TRANSFERLANGE -> D2
         MOVE.L   D2,A3
KN010:   MOVE.L   D1,D3            ;VERSCHIEBELANGE RETTEN
         ADD.L    A2,D1            ;REFE+VERSCHIEBELANGE=EMPF.-ADR.
         TRANSFER A2,D1,D2
         FILL     A2,D3,#"0"
KN012:   MOVE.W   #9,D0            ;1.SIGNIF.ZIFF.ERMITTELN
         MOVE.L   A2,A0
KN015:   CMP.B    #"0",(A0)+
         DBNE     D0,KN015
         SUB.L    A2,A0
         SUBQ.L   #1,A0
         MOVE.W   A0,D0
         MOVE.B   D0,12(A2)
KN020:   MOVEM.L  (SP)+,D1-D3/A3
         RTS
```

VORZ und die Anzahl der signifikanten Ziffern in SIGNI. Das Feld KOMMA bleibt allerdings unbeeinflußt. Die Register bleiben durch den Aufruf dieser Routine unverändert.

ASCBI

– Umwandlung von ASCII- in Binär-Format

Die Adresse von REFE muß vor dem Aufruf der Routine in A2 stehen. Das Vorzeichen bleibt unberücksichtigt. Zur Umwandlung wird die Routine MULTI benötigt. Das Ergebnis steht anschließend in D1. Alle anderen Register bleiben unverändert.

DIVI

– 32 Bit Division

Durch diese Routine wird eine 32 Bit-Division durchgeführt. Vor dem Aufruf muß der Dividend in D1 und der Divisor in D3 stehen. Nach der Ausführung findet man den Quotienten in D1, den eventuellen Rest in D2 und den Divisor in D3 wieder. Alle anderen Register bleiben unverändert.

MULTI

– 32 Bit-Multiplikation

Durch diese Routine wird eine 32 Bit-Multiplikation durchgeführt. Vor dem Aufruf muß der Multiplikant in D1 und der Multiplikator in D2 stehen. Nach der Ausführung findet man das Produkt in D1 wieder. Bis auf D1 und D2 bleiben alle anderen Register unverändert. Eine −1 in D0 zeigt einen eventuellen Überlauf an.

EDIT

– Editierroutine

Diese Routine benutzt die Macros TRANSFER und FILL. Es ist also darauf zu achten, daß sie auch im Speicher vorhanden sind. Die Adresse von REFE muß vor dem Ansprung in A2 stehen. Die zu editierende Zahl muß als ASCII-Zahl in den Feldern REFE, KOMMA und SIGNI, wie oben beschrieben, definiert sein. Außerdem ist es notwendig, das Empfangsfeld EDFLD zuvor festzulegen. Dazu muß in EDFLD die Empfangsfeldlänge und in EDFLD+1 die gewünschte Anzahl der Nachkommastellen stehen. EDFLD ist schon definiert und auf eine maximale Länge von 15 Stellen festgelegt. Die aufbereitete Zahl steht anschließend in

```
BINASC:    MOVEM.L   D0-D3/A0/A2,-(SP)
           MOVE.L    #10,D3          ;DIVISOR
           MOVE.L    A2,A0
           MOVE      #10,D0          ;11 STELLEN ERGEBNIS
BINA000:   MOVE.B    #"0",(A0)+      ;NULLEN EINFüGEN
           DBRA      D0,BINA000
           MOVE.B    #"+",-(A0)
BINA020:   JSR       DIVI
           OR.B      #$30,D2         ;REST
           MOVE.B    D2,-(A0)
           CMP.L     #0,D1           ;QUOTIENT = 0 ?
           BEQ       BINA030         ;JA, FERTIG
           JMP       BINA020
BINA030:   MOVE.W    #9,D0           ;1.SIGNIF.ZIFF.ERMITTELN
           MOVE.L    A2,A0
BINA040:   CMP.B     #"0",(A0)+
           DBNE      D0,BINA040
           SUB.L     A2,A0
           SUBQ.L    #1,A0
           MOVE.W    A0,D0
           MOVE.B    D0,12(A2)
           MOVEM.L   (SP)+,D0-D3/A0/A2
           RTS                       ;ASCII STEHT BEREIT


ASCBI:     MOVEM.L   D0/D2/D3/A0-A2,-(SP)
           MOVE.W    #9,D3           ;10 STELLEN
           MOVE.L    A2,A0
           CLR.L     D1
           MOVE.L    #10,D2
AS010:     MOVE.L    D0,-(SP)
           JSR       MULTI
           MOVE.L    (SP)+,D0
           MOVE.L    D1,A1
           CLR.L     D1
           ADD.B     (A0)+,D1
           SUB.B     #$30,D1
           ADD.L     A1,D1
           DBRA      D3,AS010
           MOVEM.L   (SP)+,D0/D2/D3/A0-A2
           RTS


DIVI:      MOVEM.L   D0/D3/D4,-(SP)
           MOVE.L    D3,D4
           MOVE      #31,D0
           CLR.L     D2
           CLR.L     D3
DIVI020:   ROXL.L    #1,D1           ;DIVIDEND IN D1
           ROXL.L    #1,D2           ;1 BIT VON D1 NACH D2 ZIEHEN
           ASL.L     #1,D3           ;QUOTIENT 1 BIT SCHIEBEN
           SUB.L     D4,D2           ;DIVISOR IN D4
           BMI       DIVI040
           ADDQ.L    #1,D3           ;QUOTIENT IN D3
DIVI030:   DBRA      D0,DIVI020
           MOVE.L    D3,D1
           MOVEM.L   (SP)+,D0/D3/D4
           RTS
DIVI040:   ADD.L     D4,D2           ;WIEDERHERSTELLEN
           BRA       DIVI030


MULTI:     MOVEM.L   D2-D4,-(SP)
           CLR.L     D3
           CLR.B     D4
           MOVE.W    #31,D0
MUL010:    ROR.L     #1,D2
           BCC       MUL020
           TST.B     D4
           BMI       MUL015
           ADD.L     D1,D3
           BCC       MUL020
MUL015:    MOVE.L    #-1,D0
           BRA       MULEXIT
MUL020:    ASL.L     #1,D1
           BCC       MUL030
           MOVE.B    #-1,D4
MUL030:    DBRA      D0,MUL010
           MOVE.L    D3,D1
           CLR.L     D0
MULEXIT:   MOVEM.L   (SP)+,D2-D4
           RTS
```

```
EDIT:      MOVEM.L   D0-D7/A0-A2,-(SP)
           MOVE.B    EDFLD,D5        ;LANGE EMPFANGSFELD RETTEN
           MOVE.B    EDFLD+1,D6      ;GEWÜNSCHTE NACHKOMMASTELLEN
           MOVE.B    #10,D0          ;LANGE VORKOMMASTELLEN
           SUB.B     11(A2),D0
           SUB.B     12(A2),D0
           CMP.B     #0,D0
           BGT       ED005
           CLR.B     D0
ED005:     EXT.W     D0              ;NACH D0
           EXT.L     D0
           MOVE.B    D6,D1           ;  WENN GEWÜNSCHT. NACHKOMMAST.
           ADDQ.B    #2,D1           ;+ LANGEN DEZIMALPUNKT U. VORZEICH.
           ADD.B     D0,D1           ;+ LANGE VORKOMMASTELLEN
           CMP.B     D5,D1           ;> FELDLANGE EDFLD, DANN
           BHI       ED030           ;  EDFLD ZU KURZ
           MOVE.B    D5,D1           ;GEWÜNSCHTE EMPFANGSLANGE
           SUB.B     D6,D1           ;- GEWÜNSCHTE NACHKOMMASTELLEN
           SUBQ.B    #1,D1           ;- 1
           EXT.W     D1
           EXT.L     D1
           ADD.L     #EDFLD,D1       ;+ ADR. EDFLD
           MOVE.L    D1,A1
           MOVE.B    #".",(A1)       ;= ADR. DEZIMALPUNKT
           MOVE.L    D1,D3           ;RETTEN
           SUB.L     D0,D1           ;- LANGE VORKOMMASTELLEN
           MOVE.L    D1,D4           ;RETTEN
           MOVE.B    12(A2),D2       ;DISPLACEMENT 1.SIGNIF.STELLE
           EXT.W     D2
           EXT.L     D2
           ADD.L     A2,D2           ;+ ADR. ERG
           TRANSFER  D2,D1,D0
           MOVE.B    #10,D1          ;10
           SUB.B     11(A2),D1       ;- NACHKOMMASTELLEN
           EXT.W     D1
           EXT.L     D1
           ADD.L     A2,D1           ;+ ADR ERG
           MOVE.L    D3,D2           ;ADR DEZIMALPUNKT
           ADDQ.L    #1,D2           ;+1 = EMPFANGSADR.NACHKOMMA -> D2
           CMP.B     11(A2),D6       ;TATSACHL.VORH./GEWÜNSCHT.NACHKOMMAST

           BGT       ED020           ;MEHR GEWÜNSCHT ALS.VORHANDEN
           MOVE.B    D6,D0
ED010:     EXT.W     D0
           TRANSFER  D1,D2,D0
           MOVE.L    D4,D0           ;EMPF.ADR. VORKOMMASTELLEN
           SUB.L     #EDFLD,D0       ;- ADR EDFLD
           FILL      #EDFLD,D0,#" "  ;FÜHRENDE BLANKS EINFÜLLEN
           CMP.B     #"-",10(A2)     ;NEGATIVES VORZEICHEN ?
           BNE       ED999           ;NEIN
           MOVE.L    D4,A0           ;EMPF.ADR.VORKOMMASTELLEN
           SUBQ.L    #1,A0           ;- 1
           MOVE.B    #"-",(A0)       ;= VORZEICHENSTELLE
ED999:     MOVEM.L   (SP)+,D0-D7/A0-A2
           RTS
ED020:     MOVE.B    D6,D0           ;  GEWÜNSCHTE NACHKOMMASTELLEN
           SUB.B     11(A2),D0       ;- VORHANDENE
           EXT.W     D0              ;= MIT NULLEN AUFZUF.STELLEN
           MOVE.B    11(A2),D7       ;VORHANDENE NACHKOMMASTELLEN
           EXT.W     D7
           EXT.L     D7
           ADD.L     D2,D7           ;+ EMPF.ADR.NACHKOMMASTELLEN
           FILL      D7,D0,#"0"      ;NACHKOMMASTELLEN AUFFÜLLEN
           MOVE.B    11(A2),D0       ;IN TATSACHL.VORH.LANGE
           BRA       ED010           ;ÜBERTRAGEN
ED030:     FILL      #EDFLD,D5,#"#"  ;AUFFÜLLEN MIT #, DA ZU KURZ
           BRA       ED999
ENDSTAND:  EQU       *
```

der geforderten Länge rechtsbündig mit führenden Leerstellen und Dezimalpunkt in EDFLD. Dabei werden nur negative Vorzeichen eingefügt, positive nicht. Wenn die Länge von EDFLD zu kurz gewählt wurde und deshalb die aufzunehmende Zahl zu groß ist, werden #-Zeichen angezeigt.

Man muß also EDFLD so lang wählen, daß mindestens die Ziffernanzahl, der Dezimalpunkt und das Vorzeichen Platz haben. Erwartet man zum Beispiel Zahlen bis 1000, muß mindestens eine Sechs vor Aufruf der Routine in EDFLD stehen. Alle Register bleiben unverändert.

Wir haben nun den PRIMA-Teil von ISAM & PRIMA abgeschlossen und wenden uns im nächsten Heft dem letzten, dem ISAM-Teil zu. Dies dürfte insbesondere für alle, die sich mit Dateierstellung beschäftigen wollen, sehr interessant werden.

(HE)



T. E. Shoup
Numerische Verfahren

Carl Hanser Verlag, München
1. Auflage 1985
225 Seiten
DM 48,–

Das Buch beschäftigt sich ausschließlich mit Lösungsverfahren für algebraische und transzendente Gleichungen (z. B. Newton-Methode), Gleichungssystemen (Gauss-Jordan/Seidel-Verfahren), Eigenwertproblemen (Tridiagonalmatrix, Hessenbergform), Differentialgleichungen, Interpolation (linear, Lagrange, iterative, invers), Kurvenanpassung (Methode der kleinsten Quadrate, Spline-Funktion), Differentiation (numerisch) und Integration (Simpson, Romberg, Gauss). Damit ist ein breites Spektrum für den technisch-wissenschaftlichen Bereich abgedeckt.

Alle Programme sind in BASIC geschrieben und deshalb einfach auf andere Sprachen anzupassen. Viele würden sich vielleicht eine andere Programmiersprache wünschen, doch die Algorithmen sind nicht so kompliziert oder lang, daß dies ins Gewicht fallen würde.

Jedes Verfahren wird leichtverständlich erläutert und von Herleitungen und Grafiken unterstützt. Ein konkretes Beispiel mit Zahlenwerten zum Überprüfen des Programms fehlt ebenfalls nicht.

Das Buch behandelt in lockerer Form wichtige Verfahren, ohne dabei den Leser zu überfordern oder zu langweilen.

(mn)

Niklaus Wirth
Algorithmen und Datenstrukturen

B. G. Teubner, Stuttgart
3. überarb. Aufl. 1983 (in Pascal)
4. neubearb. Aufl. 1986 (in Modula-2)
ca. 300 Seiten
DM 38,–

Das bereits 1975 erschienene Buch von Niklaus Wirth zählt, in Bezug auf dynamische Datenstrukturen und Sortieralgorithmen, zu einem der Standardwerke. Bis zur dritten Auflage sind die Beispiele dabei ausschließlich in **PASCAL** formuliert, die gerade erschienene 4. Auflage dagegen in **MODULA-2**. Sie wurde nochmals erweitert, entspricht jedoch im wesentlichen der hier vorliegenden Version.

Das Buch beginnt mit einer Einführung in die grundlegenden Datenstrukturen ARRAY, (varianter) RECORD, SET und (sequentielle) FILES.

Diese werden im vierten Kapitel noch um die erheblich leistungsfähigeren, dynamischen Strukturen erweitert. Dazu gehören Zeiger, Lineare Listen, Schlüssel-Transformationen und vor allem die Baumstrukturen (Binär- und Vielwegbäume). Baumstrukturen sind zum Sortieren und nachfolgenden Suchen sehr gut geeignet, wobei der Optimierung und der Erstellung von ausgeglichenen Bäumen eine besondere Bedeutung zukommt. Das Buch liefert die hierfür erforderlichen Pascalprogramme und außerdem die Routinen zum Löschen, Einfügen und Durchsuchen.

Die z. T. recht komplexen und damit auch komplizierten Verfahren werden anhand von etlichen Diagrammen verdeutlicht.

Im Kapitel über das Sortieren, wird eine große Anzahl von Strategien beschrieben und deren Vor- und Nachteile erläutert. Die Spanne reicht vom einfachen Bubble- und Shakersort über direktes/binäres Einfügen, direkte Auswahl, Shell- und Heap- und Mergesort bis zum schnellen Quicksort (rekursiv und iterativ). Jeder Algorithmus wird ausführlich vorgestellt, wozu auch ein Probelauf und eine Berechnungsformel für die Anzahl der Operationen gehört. Den Abschluß bildet ein Zeitvergleich der vorgestellten Verfahren unter verschiedenen Bedingungen: geordnet, zufällig, umgekehrt sortiert und bei mächtigeren Daten. Der zweite Teil des Sortierkapitels behandelt Verfahren, die benötigt werden, wenn die Datenmenge nicht in den Hauptspeicher des Rechners paßt. Vorgestellt werden: Direktes Mischen, natürliches Mischen, ausgeglichenes n-Weg-Mischen und das Mehrphasen-Sortieren.

Die 4. Auflage enthält, als Ergänzung zu den Sortierverfahren, noch verschiedene, schnelle Suchalgorithmen (Binäre- und Tabellensuchen, Mustersuchen in Zeichenketten, Knuth-Morris-Pratt- und Boyer-Moore Algorithmus), die zur Zeit der Erstauflage noch weitgehend unbekannt waren.

Das letzte Kapitel ist rekursiven Programmen gewidmet. Hier werden u.a. zwei graphisch interessante Routinen erklärt: die überlagerten Hilbert- und Sierpinsi-Kurven. Auch das bekannte Acht-Damen-Problem wird mit Hilfe einer rekursiven Prozedur gelöst.

Jedes Kapitel wird mit einer Anzahl von Aufgaben abgeschlossen, die in erster Linie als Anregung für eigene Programmierübungen dienen sollen, denn die Lösungen werden nicht angegeben.

„Algorithmen & Datenstrukturen“ ist ein wirklich interessantes Buch, das dem interessierten Programmierer eine wertvolle Hilfe und Programmbibliothek (Sortierprogramme, Baumstrukturen) ist.

(mn)

DER ST KALENDER 1987

Pünktlich zum neuen Jahr bringen wir Ihnen zwölf Bilder, die auf dem ATARI ST entstanden sind, als ST-Kalender. Sie wurden mit verschiedenen Mal- oder Zeichenprogrammen von fleißigen ST-Künstlern produziert.

Wir haben uns die Auswahl nicht leicht gemacht. Zuerst stellte sich die Frage, ob wir Farb- oder Monochrombilder in den Kalender aufnehmen wollten. Wir entschieden uns für beides. Einige Leser schickten Digitizerbilder ein, die wir nach langer Diskussion doch nicht in die Wertung miteinbeziehen wollten, da ja nicht im eigentlichen Sinne gezeichnet wurde. Wir spielen allerdings mit dem Gedanken, eine ST-Kunstgalerie in der „ST Computer" einzurichten, in der jeden Monat eines oder mehrere Bilder abgedruckt werden. Die Entscheidung darüber hängt allerdings von der Zahl der Einsendungen ab. Kommt die Galerie zustande, ist natürlich jedes Hilfsmittel, egal ob Digitizer, Grafiktablett oder Maus zugelassen. Digitizerbilder müßten allerdings auch noch von Hand bearbeitet werden, damit ein Motiv nicht nur mit einem Tastendruck und der nötigen technischen Ausrüstung erstellt wird.

Das nächste Problem stellte sich uns mit Garfield-, Werner- und ähnlichen Bildern, die mit einem Copyright geschützt sind. Sie konnten wir leider nicht veröffentlichen, obwohl sehr schöne Bilder darunter waren. Sorry!

Viele Leser werden unsere Bildauswahl für zu konservativ halten. Wir waren jedoch der Meinung, daß moderne Grafikbilder gegenüber den restlichen Bildern ein Stilbruch wären. Hätten wir genügend Einsendungen in dieser Stilrichtung gehabt, wäre wahrscheinlich noch ein zweiter Kalender veröffentlicht worden.

Trotzdem bitten wir alle Einsender, die nicht zum Kreis der „Auserwählten" gehören, nicht enttäuscht zu sein – aber es konnte eben für jeden Monat nur ein Bild abgedruckt werden.

Und wir rufen Sie gleich zum erneuten Zeichnen auf, da wir – genügend Einsendungen vorausgesetzt – bereits den ST-Kalender für das Jahr 1988 planen. Der soll dann nicht nur in der Januar-Ausgabe '88 abgedruckt werden, sondern auch als richtiger Wandkalender erscheinen.

(HE)

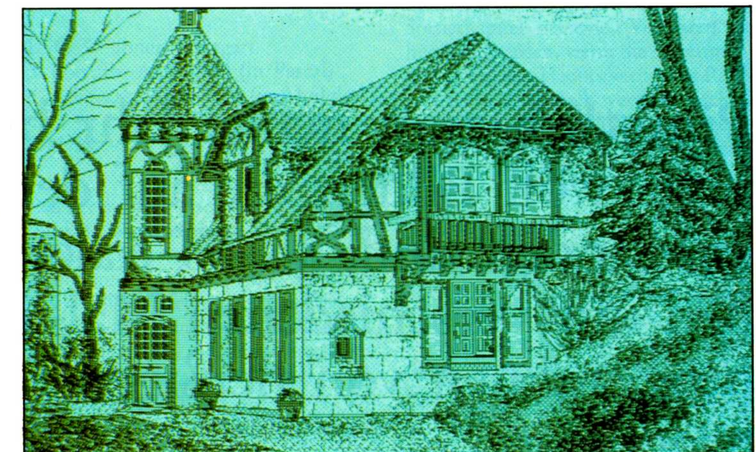
J A N U A R

			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	31	



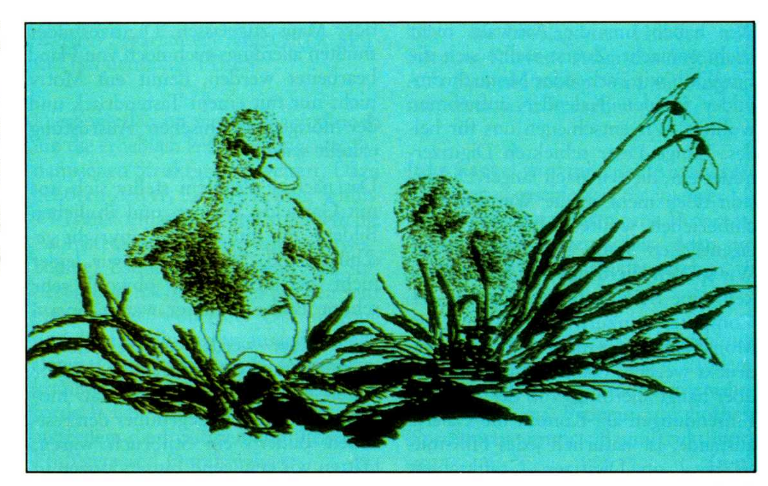
F E B R U A R

						1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	



M Ä R Z

						1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30	31					





...UND PLÖTZLICH HAT IHR ATARI ST DAS LAUFWERK DER PROFIS



z. B. vortex MA1-D

Ein 3.5" Doppellaufwerk der jüngsten Generation mit 1,4 MB formatierter Speicherkapazität. **Kompakt:** 290 (L) × 105 (B) × 64 (H) mm. **Komplett:** Integriertes Netzteil (25 W, 220 V, 50 Hz). **Kompatibel:** Atari-gleicher Floppy-Stecker; Atari-gleiche Lackierung. **Kostengünstig:** Nur 998,– DM*. Den vortex MA1 gibt es auch als Single-Laufwerk zum späteren Aufrüsten. Außerdem im vortex-Atari-Tuning-Programm: 5.25"-Einzel- und Doppellaufwerke. 3.5"- plus 5.25"-Laufwerk („Gemischtes Doppel").

vortex verwendet NEC- bzw. BASF-Laufwerke.

*empfohlener Verkaufspreis.

I·N·F·O-S·C·H·E·C·K

Bitte senden Sie mir weitere Informationen über Ihre „Atari"-Laufwerke und einen Händlernachweis.

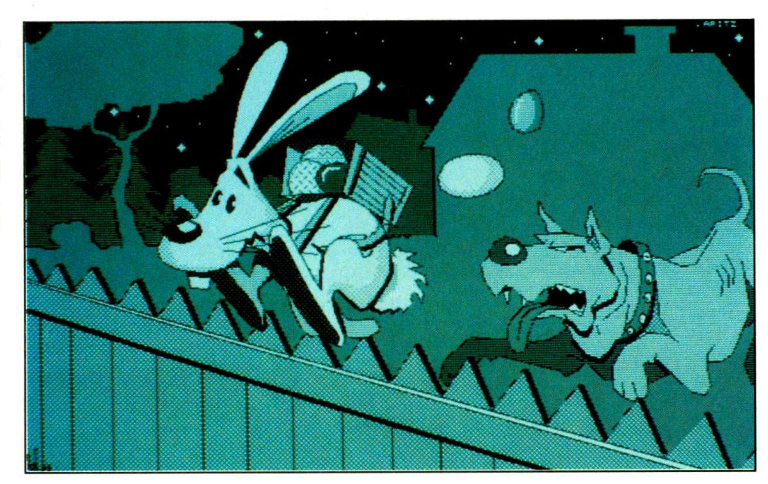
vortex Computersysteme GmbH · Falterstraße 51-53 · 7101 Flein

vortex COMPUTERSYSTEME

...UND PLÖTZLICH HABEN SIE EINEN PROFI-COMPUTER

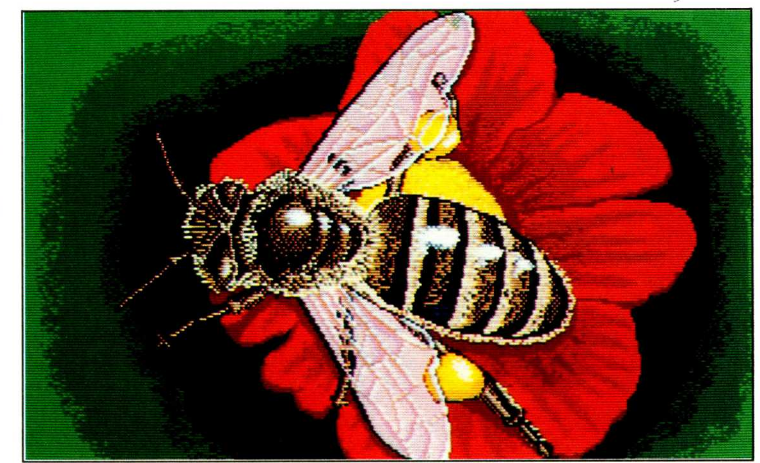
A P R I L

		1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30			



M A I

				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	31



J U N I

1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30					



...UND

PLÖTZLICH

ERREICHEN

SIE

MIT IHREM

ATARI ST

UNGEAHNTE

DIMEN-

SIONEN

Mit der vortex HD20-Station.



Ein Hard-Disk-Laufwerk („Winchester") mit 21 MB formatierter Speicherkapazität. **Das sind rund 10.000 vollgeschriebene DIN A 4-Seiten!** Dazu der blitzschnelle Zugriff: 85 ms! Die vortex-HD20 ist komplett und kompakt: Netzteil (32 W, 220 V), „Winchester"-Laufwerk und HOST-Adapter sind in einem Atari-weißen Gehäuse (L 300 mm x B 110 mm x H 64 mm) untergebracht. Systemdiskette und ausführliches deutsches Handbuch werden mitgeliefert. **Machen Sie aus Ihrem Atari ST einen echten Profi-Computer. Nutzen Sie unser Test-Angebot.**



BESTELL-COUPON

☐ Senden Sie mir die vortex-HD20-Station versandkostenfrei per Nachnahme. Ich kann sie ohne Angabe von Gründen innerhalb 10 Tagen zurücksenden und erhalte den Kaufpreis in Höhe von DM 1.798,– umgehend erstattet.

☐ Ich möchte nähere Informationen über die vortex-HD20-Station.

Name ____________________

Anschrift ____________________

Unterschrift ____________________

Datum ____________________

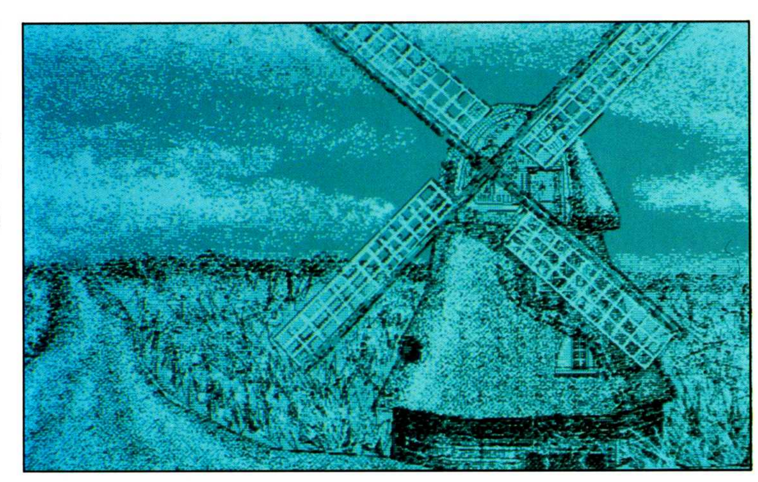
vortex Computersysteme GmbH · Falterstraße 51-53 · 7101 Flein



...UND PLÖTZLICH HABEN SIE EINEN PROFI-COMPUTER

J U L I

		1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30	31		



A U G U S T

					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
31						



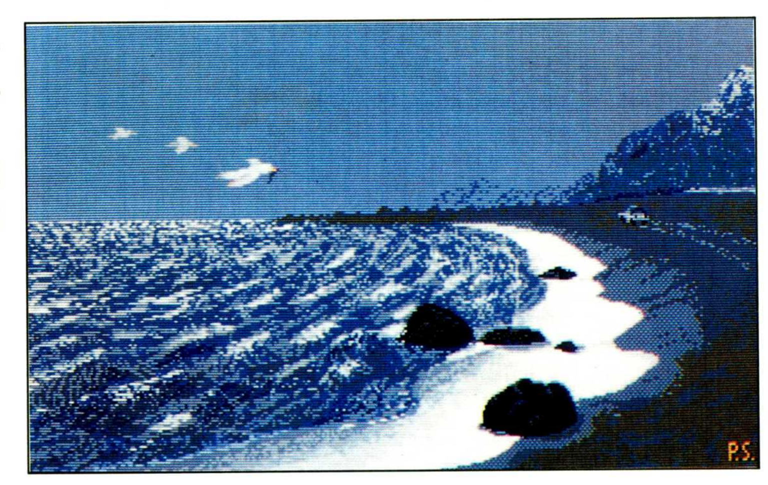
S E P T E M B E R

	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30				



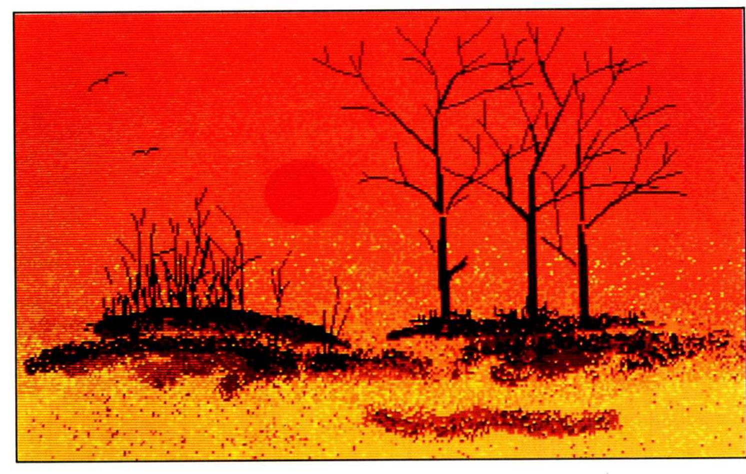
OKTOBER

			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	31	



NOVEMBER

						1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30						



DEZEMBER

	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	31			







FLIGHT SIMULATOR II

Endlich gibt es ihn auch für den ATARI ST. SUBLOGIC hat die Fans lange warten lassen, doch dafür hat er einige Features bekommen, welche die Versionen für IBM, APPLE u. a. noch übertreffen.

Die Graphik ist dank des 16-Bit-Prozessors recht schnell und gleitet fast ruckfrei über das Bildschirmfenster, dessen Größe und Lage vom Benutzer bestimmt werden kann. Es kann sehr klein sein oder auch den gesamten Bildschirm ausfüllen. Dann sind jedoch die Instrumente nicht mehr sichtbar - doch wofür gibt es den Auto-Piloten!

Die Blickrichtung ist in allen drei Ebenen frei wählbar, ohne die Flugrichtung zu verändern. Besonders schön ist in diesem Zusammenhang die Zoomfunktion, die so weit geht, daß ein winziges Objekt den ganzen Bildschirm(!) ausfüllen kann. Sie wirkt sich auch auf die verschiedenen Blickpunkte aus, von denen der Flug verfolgt werden kann. Neben der 'normalen' Sicht aus dem Cockpit, kann der Blick vom Tower und oberhalb des Flugzeugs (Landkartenansicht) simuliert werden. Zwei dieser Ansichten können sogar gleichzeitig dargestellt werden, z. B. nebeneinander oder auch überlagert.

Die Kontrolle der wichtigsten Steuerungen (Geschwindigkeit, Richtung, Bremsen) erfolgt mit der Maus, die Zusatzfunktion über die Tastatur. Fast alle Tasten sind mit Funktionen belegt, die übersichtlich in einer Graphik dargestellt sind. Diese Graphik ist zum Steuern des Flugsimulators, zumindest am Anfang, unbedingt erforderlich, denn niemand kann sich so viele Funktionstasten auf einmal merken.

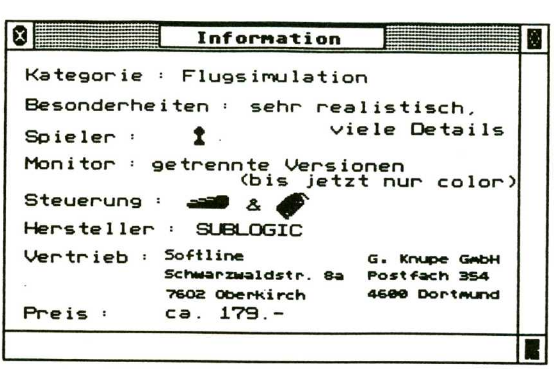
Auch die Menge der Instrumente - Höhenmesser, Geschwindigkeitsmesser, künstlicher Horizont und wie sie alle heißen - und Anzeigen überfordert den unerfahrenen Flieger anfangs völlig, doch nach dem Studium des Handbuchs ist vieles klarer.

Neben der zum Üben gut geeigneten **Cessna 182**, gibt es noch den schnellen **Learjet 25G**, der schon erheblich mehr können erfordert. Zum Steigern des Schwierigkeitsgrades gibt es neben dem Tagflug noch den Dämmerungs- und Nachtflug, außerdem kann das Wetter geändert werden; auf Wunsch erscheinen Wolkenschichten, Nebel, Wind und Turbulenzen. Wem das noch nicht reicht, der kann sich im Kunstflug probieren und Rollen oder Loopings fliegen.

Zum Flugsimulator gehört ein umfangreiches englisches Handbuch (mit vielen guten Zeichnungen), **sechs Landkarten** mit den Koordinaten der jeweiligen Flughäfen (Gebiete: New York, Chicago, Seattle, Los Angeles, San Francisco; insgesamt 120), eine Karte mit den Maus- und Tastenbelegungen und eine **deutsche Kurzanleitung**, die sich jedoch nicht speziell auf die ST-Version bezieht.

Ausgeliefert wird FLIGHT SIMULATOR II in zwei getrennten Versionen, eine für Farb- und eine für SW-Monitore. Der Preis liegt mit DM 179,- an der oberen Preisgrenze für ein Spiel. Einen Fan wird dies jedoch kaum abschrecken, denn man bekommt ein sehr ausgereiftes und bewährtes Produkt, das durch seine Variationsvielfalt keine Zeit für Langeweile aufkommen läßt.



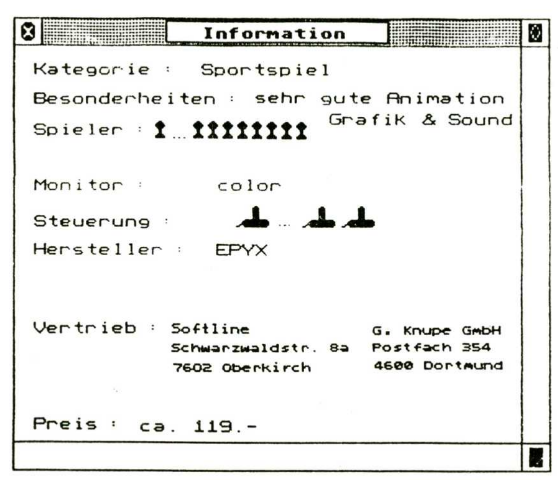


WORLD GAMES

Eine spannende Weltreise erlebt man mit WORLD GAMES, den neuen 'olympischen Spielen' des Software-Hauses EPYX. Jede Disziplin steht dabei stellvertretend für ein anderes Land.

Es beginnt in Rußland, der Heimat der besten Gewichtheber der Welt. In den Disziplinen 'Snatch' und 'Clean and jerk' wird der Wettkampf ausgetragen und es kommt auf Geschicklichkeit und vor allem Timing an. Die Zeiten zwischen den einzelnen Bewegungen müssen genau eingehalten werden - bei zu schneller Ausführung hat der Athlet noch nicht genug Kraft gesammelt. Wenn man zu lange wartet, dann hat sie ihn schon wieder verlassen. Vor über 300 Jahren kamen europäische Schlittschuhläufer auf die Idee über Fässer zu springen. Diese sehr gefährliche Sportart erfordert eine hohe Geschwindigkeit und einen Hang zum Risiko beim Absprung, denn wer zu früh abspringt verschenkt wertvolle Meter und wer zu lange zögert, der stürzt gleich ins erste Faß.

An den Felsen des sonnigen Acapulco findet die nächste, nicht minder gefährliche Disziplin statt - das Klippenspringen. Nur wer hier richtig abspringt, der trifft nicht auf die unteren Klippen. Auch der Zeitpunkt des Absprungs ist nicht unbedeutend, da sich der Wasserspiegel ständig hebt und senkt.



Das bekannte Wintersportgebiet Chamonix in Frankreich ist der Schauplatz des Skislaloms. Die Geschwindigkeit spielt hier weniger eine Rolle, als das sichere Umfahren (nicht Umfahren!), der Stangen. Eine einzige unbedachte Bewegung wirft den Spieler schon in den Schnee und damit aus dem Spiel.

Canada: zwei Holzfäller stehen auf einem Baumstamm, der im Wasser schwimmt. Jeder versucht, durch Hin- und Herdrehen des Stammes, den Gegner ins eisige Wasser zu werfen. Es gilt, einen guten Rhythmus zu finden, die Balance zu halten und schnell zu reagieren.

Aus dem amerikanischen Westen kommt, als Abart des Rodeos, das Bullenreiten. Nur wem es gelingt, sich acht Sekunden auf dem Bullen zu halten, der erhält eine Wertung. Diese bezieht auch den Stil des Reiters und die Wildheit des Tieres mit ein. Wer hohe Noten erreichen will, wählt 'Earthquake', den wildesten und gemeinsten Bullen, aber dann muß jede dessen Bewegungen ausgeglichen werden, sonst landet man im Staub der Arena.

Schottische Dudelsackmusik: als bekannteste Disziplin der 'Highland Games' wurde das Baumstammwerfen ausgewählt. Rhythmisch Anlauf nehmen, in die Hocke gehen und den Stamm werfen. Wenn der Stamm nach der Landung in die andere Richtung kippt, dann ist ein gültiger Wurf geglückt. Fällt er dagegen rückwärts, dann besteht die Gefahr, daß er auf dem eigenen Fuß oder Kopf landet. Ein gelungener Wurf wird mit einer neuen Melodie auf dem Dudelsack gewürdigt - da lohnt sich das Trainieren!

Sumo ist eine traditionsreiche japanische Sportart, bei der sich zwei schwergewichtige, dickbäuchige Ringer gegenüberstehen. Nach der Begrüßungszeremonie legen die, oft bis zu 400 Pfund schweren Kerle los. Sechzehn verschiedene Griffe sind möglich, um den Gegner aus dem Ring zu drücken oder auf den Rücken zu legen - keine leichte Aufgabe. Schnelligkeit und Timing sind auch hier wieder die entscheidenden Kriterien für hohe Punktzahlen.

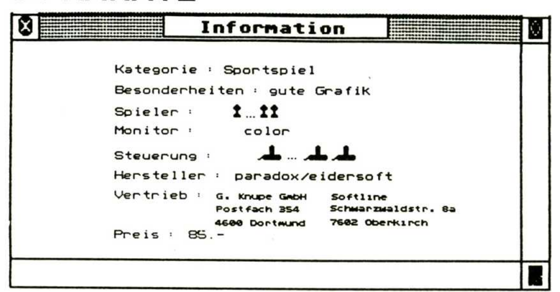
Ein wirklich reichhaltiges Programm, daß man auf dem Weg zum Champion zu bewältigen hat. Bei überragenden Leistungen wird man sogar in die Hi-Score-Liste übernommen, die 'selbstverständlich' abgespeichert wird.

WORLD GAMES gehört, wie auch schon WINTER GAMES, zu den besten Sportspielen auf dem ST.





ST-KARATE

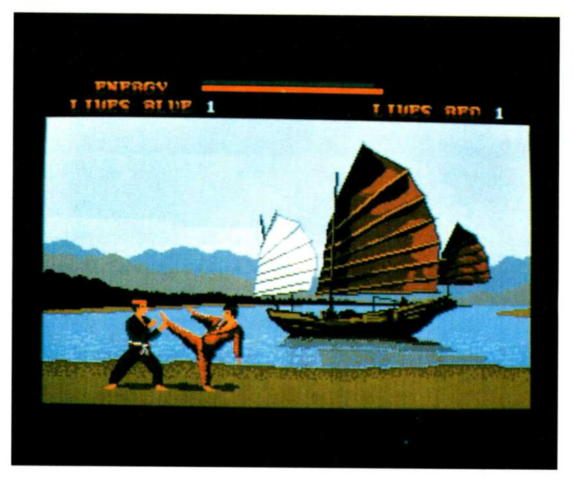

Information

Kategorie : Sportspiel
Besonderheiten : gute Grafik
Spieler : 1 ... 2
Monitor : color
Steuerung : 1 ... 2
Hersteller : paradox/eidersoft
Vertrieb : G. Knupe GmbH, Postfach 354, 4600 Dortmund — Softline, Schwarzwaldstr. 8a, 7602 Oberkirch
Preis : 85.-

Markerschütternde Schreie hallen durch die Luft - einige dumpfe Schläge folgen - etwas fällt plump auf den Boden - der Kenner weiß jetzt schon, daß es sich hierbei um das allseits beliebte Karate-Spiel handelt.

Vor einmalig schönen asiatischen Kulissen findet der Kampf zwischen zwei Karatekas statt. Sie wirbeln durch die Luft, schlagen und treten um sich und plötzlich liegt einer von ihnen am Boden. Das Szenenbild wechselt und weiter geht der Kampf.

Die Kraft jedes Spielers wird diesmal in Form eines Balkens dargestellt, dessen Länge bei jedem erhaltenen Treffer abnimmt. Wenn er aufgebraucht ist, bricht der Karateka zusammen. Jeder Spieler hat jedoch fünf Karatekas und erst, wenn alle geschlagen sind, ist der Kampf verloren. Die einzige Möglichkeit, Kraft dazuzubekommen ist, einen manchmal



auftauchenden hüpfenden Krug zu zertreten. Dieser Kraftzuwachs ist u. U. spielentscheidend.

Wer hier weiterkommen will muß schon etwas üben, denn ganz so einfach ist die Steuerung mittels Joystick nicht. Immerhin sechzehn verschiedene Bewegungen können damit bewirkt werden.

ST-KARATE ist ein sehr amüsantes und fesselndes Spiel, das man immer wieder gerne spielt. Die Grafik zählt zum Besten was derzeit geboten wird.



JOUST

Information

Kategorie : Actionspiel
Besonderheiten : Kooperation zweier Spieler
Spieler : 1 ... 2
Monitor : color
Steuerung : 1 ... 2
Hersteller : ATARI
Vertrieb : G. Knupe GmbH, Postfach 354, 4600 Dortmund
Preis : 98.-

In der kämpferischen Atmosphäre eines mittelalterlichen Turnierspiels sieht sich ein einsamer Ritter auf einem Vogelstrauß einer Unzahl, gleichfalls fliegender, Gegner gegenübergestellt.

Die Steuerung des Vogelstrauß ist sehr witzig, denn er ist nur durch ständiges Drücken des Feuerknopfes in der Luft zu halten. Ansonsten sinkt er rasch ab, und ehe man sich versieht, ist er der glühenden Lava zu nahe gekommen, oder ein gegnerischer Ritter hat ihn im Zweikampf besiegt.

Dies ist nicht besonders tragisch, und man ist bei solchen Spielen daran gewöhnt, daß sich nach kurzer Zeit ein neuer Ritter aus dem Nichts kristallisiert. Wer jedoch allzu lässig mit seinen Rittern umgeht, der wird die höheren Level nicht erreichen, denn dort warten noch einige Überraschungen. Ab dem vierten Level greift die Hand des Lava-Trolls nach tieffliegenden Kämpfern und versucht, sie nach unten zu ziehen. Außerdem erscheint immer öfter ein prähistorischer Flugsaurier namens 'Pterodactyl', der zielstrebig auf den eigenen Ritter zufliegt. Dieser Zeitgenosse ist wirklich sehr unliebsam und zudem auch noch unverwundbar.

In einem speziellen Modus können zwei Spieler gleichzeitig am Geschehen teilnehmen und gemeinsam Punkte erringen. Dies ist eine besonders knifflige Angelegenheit, weil man im Eifer des Gefechts leicht den Mitspieler 'vom Vogel holt'. Hier zeigt sich, wer seinen Vogel am besten manövrieren und dessen verzögerte Abbremsung am besten timen kann.

Die Spielmotivation von JOUST ist dann sehr hoch, wenn man der Jagd nach dem 'High-Score' verfällt, der glücklicherweise auch auf der Diskette abgespeichert wird. Das Spielgeschehen erfordert Konzentration, Reaktionsschnelligkeit und nicht zuletzt einen durchtrainierten Daumen, um den Vogel in der Luft zu halten. Insgesamt ist das Spiel sehr lustig und spannend und kann deshalb auch empfohlen werden.

Profitieren Sie von unserem Know-How!

OMIKRON-BASIC



ARITHMETIK
- Rechengenauigkeit bis 19 Stellen bei allen Funktionen
- Rechenbereich bis 5.11 E±4931

GEM
- Komplette GEM-Library
- sämtliche AES- & VDI-Funktionen direkt mit Namen verfügbar
- eigene BITBLIT-Routine
- NEU: jetzt mit Resource Construction Set

EXTRAS
- Masken-INPUT
- SORT-Befehl sortiert beliebige Felder
- Matrizenbefehle
- Compiler ab Frühjahr verfügbar

STRUKTUR
- Prozeduren und mehrzeilige Funktionen
- mit Übergabe- und Rückgabe-Parametern und lokalen Variablen
- REPEAT...UNTIL, WHILE...WEND, mehrzeiliges IF...THEN...ELSE...ENDIF
- Labels bei GOTO, GOSUB, ON...GOTO etc.

KOMPATIBILITÄT
- 99 % MBASIC-kompatibel
- Editor findet Inkompatibilitäten
- dadurch einfachste Anpassung

GESCHWINDIGKEIT
- FIT-Code (FIT = Fast Interpreting Technique)
- Wir kennen keine schnelleren 68000-Fließkommaroutinen
- Volle Integer-Arithmetik
- eigene Disk-Routinen für beschleunigten Dateizugriff

LIEFERUMFANG
- Modul mit OMIKRON-BASIC (wird seitlich eingesteckt)
- Demodiskette mit Runtime-Interpreter und Hilfsprogrammen
- 170-seitiges, spiralgebundenes deutsches Handbuch
- Alles zusammen nur DM 229,–
  Handbuch vorab DM 30,–

„Atemberaubende Geschwindigkeit" (DATA WELT 12/86, S. 69)
„Für ernsthafte Programmierer, die effektive und vor allem schnelle Programme erstellen wollen, ..., ist dieser neue BASIC-Interpreter genau das richtige" (DATA WELT 12/86, S. 70)
„OMIKRON-BASIC kann wesentlich mehr als der BASIC-Standard" (ST Computer 12/86, S. 79)

OMIKRON Software · Erlachstr. 15 · 7534 Birkenfeld · Tel. (0 70 82) 53 86

PADERCOMP – Walter Ladz

Erzbergerstr. 27 · 4790 Paderborn · Tel. 0 52 51 - 3 63 96

FLOPPYSTATIONEN FÜR ATARI ST®

PADERCOMP FL 1 .... **448,–**
3.5", 1 Mb, eingeb. Netzteil, NEC-Laufwerk, Abm. 240x105x40 mm, anschlußfertig mit Industrie Floppystecker, graues Metallgeh. ohne Schrauben an den Seiten.

PADERCOMP FL 2 .... **798,–**
Doppellaufwerk übereinander, sonst wie FL 1

NEC FD 1036 A 3.5", 1 MB, 32 mm Bauhöhe **269,–**

dto. ST modifiziert **289,–**

Industrie Floppystecker .... **nur 9,90**

ST Kabel an Shugart-Bus 3.5" **29,90**

ZUBEHÖR

3.5" Disketten .... **Superpreise!**

Disk Box SS-50, f. 50 3.5" Disketten **24,90**

Druckerkabel ST **34,90**

Dataphon S21/23, 300 bzw. 1200/75 Baud, BTX **338,–**

CDI-Hitrans 300c, 300 Baud, vollduplex, mit FTZ-Nr. **239,–**

Orion Farbmonitor CCM 1280 m. Kabel an Atari 260/520 **888,–**

Philips Farbmonitor CM 8524 .... **nur 539,–**

NEC Multisync, alle drei Auflösungen .... **Sonderpreis**

Monitor-Ständer dreh-, schwenk- und kippbar **35,–**

Preisliste .... **kostenlos**

DRUCKER

STAR NL 10 incl. Interface **719,–**
Citizen 120D der Einsteigerdrucker **579,–**
OKIDATA ML 192 incl. vollaut. Einzelblattein. 249,. **1398,–**
Panasonic KX-P 1080, 100 Z/s, NLQ **648,–**
Panasonic KX-P 1091, 120 Z/s, NLQ **768,–**
Panasonic KX-P 1092, 180 Z/s, NLQ **1049,–**

Ein Schriftbild, fast wie gesetzt!
24-Nadeldrucker von NEC

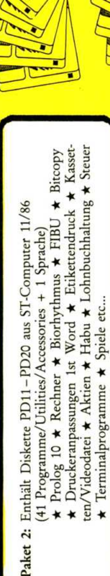
NEC P6, 24 Nadeln, 216 Zeichen, DIN A 4 .... **1438,–**
NEC P7, 24 Nadeln, 216 Zeichen, DIN A 3 .... **1828,–**
NEC P5, 24 Nadeln, 264 Zeichen, DIN A 3 .... **2898,–**

Bestellungen per Nachnahme oder Vorkasse ab 30 DM. Auslandslieferungen nur gegen Vorkasse. Eingetragenes Warenzeichen: ATARI ST. Die Preise können günstiger liegen. Rufen Sie an! Händleranfragen erwünscht.

Public-Domain Software Weihnachtspakete

★ **5 Markendisketten (doppelseitig formatiert) gefüllt mit guter Public-Domain Software** — Paketpreis nur DM **45,–**

★ **10 Markendisketten (einseitig formatiert) gefüllt mit guter Public-Domain Software** — Paketpreis nur DM **78,–**











Paket 1: Enthält Diskette PD01–PD10 aus ST-Computer 11/86 (31 Programme/Utilities/Accessories + 2 Sprachen) ★ Malprogramme ★ Diskmonitore ★ Formatter ★ Kartei-Kasten ★ komplettes Forth System ★ Ram Disk ★ Copy Programme ★ Programmiersprache LISP ★ Mini CAD ★ Spiele etc...

Paket 2: Enthält Diskette PD11–PD20 aus ST-Computer 11/86 (41 Programme/Utilities/Accessories + 1 Sprache) ★ Prolog 10 ★ Rechner ★ Biorhythmus ★ FIBU ★ Bitcopy ★ Druckeranpassungen 1st Word ★ Etikettendruck ★ Kassetten/Videodatei ★ Aktien ★ Habu ★ Lohnbuchhaltung ★ Steuer ★ Terminalprogramme ★ Spiele etc...

Paket 3: Enthält Diskette PD21–PD30 aus ST-Computer ★ Haushalt ★ Dateiverwaltung ★ Adressverwaltung ★ Plotter ★ Terminalprogramm ★ Vokabeltrainer ★ Rescratch ★ Spiele ★ ST-Sound Demo ★ GfA-Basic Programme etc...

Paket 4: Aus Kalifornien eingetroffen, enthält ★ Textprogramme (z. B. auch als Deskaccessories) ★ Fonts ★ Ramdisks ★ Graphics ★ Robots Tos ★ Colors ★ viele Spiele ★ Utilities ★ CAD ★ Recover Tos ★ u.v.m... **lassen Sie sich überraschen!!!**

Paket 5: Enthält 3 Ausgaben **ST News**, über 230 Seiten Infos, Tips und Programmlistings. Weiterhin 9 Disketten aus Kalifornien, z. B. NEO2Degas, Printdir, Disked.TOS, Dungeon, Memtest.TOS, Eliza.Tos, Sound + Sound2, Tinytool.C, Crash dump, **Mailmerge**, Tutorial.GEM uvm. **Die Fortsetzung von Paket 4!**

Paket 6: Enthält weitere 10 Disketten aus aller Welt, mit Schwerpunkt Nordamerika. Z. B. Memory Map, Filedate FTH, Compare FTH, Quiz TOS, Blue ACC, Mickey, Pasfix1, Columbus, Palette ACC, etc...

Ja, ich möchte folgende Public-Domain Weihnachtspakete bestellen.

Menge	Bitte ankreuzen ob doppelseitig oder einseitig formatiert Paketpreis Paketpreis	Gesamtpreis
☐ Paket 1	☐ 45,– oder ☐ 78,–	
☐ Paket 2	☐ 45,– oder ☐ 78,–	
☐ Paket 3	☐ 45,– oder ☐ 78,–	
☐ Paket 4	☐ 45,– oder ☐ 78,–	
☐ Paket 5	☐ 45,– oder ☐ 78,–	
☐ Paket 6	☐ 45,– oder ☐ 78,–	

☐ Scheck über DM ____________ habe ich beigefügt
☐ Lieferung per Nachnahme (zzgl. DM 5,– Porto und Verpackung)

Anschrift:

IDL Software, 1393 Green Valley Rd., Napa CA. 94558

Bestellungen einsenden an:

Agentur Nathusius „Public Domain“ Postfach 418 6140 Bensheim 1

Adimens ST:

Eine schnelle relationale Datenbank für den Atari ST

Datenbanksysteme sind heute ein fester Bestandteil der Microcomputer-Anwendungen. Sie ersetzen die alten Karteikästen: Sei es in privatem oder im industriellen Bereich. Doch was ist überhaupt ein „relationales Datenbank-System"?

Der Begriff „relational" kommt aus der Mathematik. Danach ist eine Relation eine Tabelle mit gleichartigen Zeilen, die nach einem Schlüssel sortiert sind. In der Datenverarbeitung ist eine Datei mit mehreren gleichartigen Datensätzen eine solche Relation. Zum Beispiel:

Name	Vorname	Telefon	Kunden-Nr.
Müller	Hans	0815/4711	6001
Heim	Karl-Heinz	06151/56057	6007

Man kann mehrere Datensätze über einen gemeinsamen Schlüssel verknüpfen – ein Schlüssel definiert einen Datensatz eindeutig – und ihre gegenseitige Verknüpfung in zweidimensionalen Tabellen darstellen.

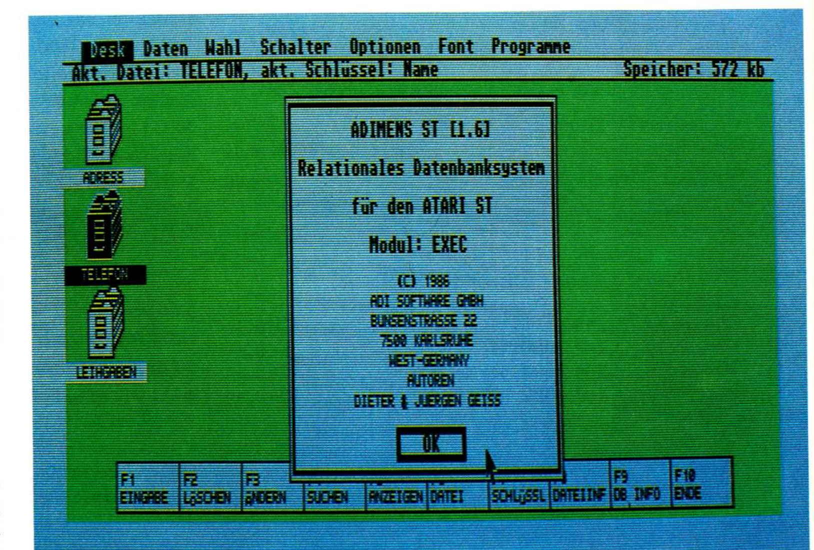
Logische Beziehungen zwischen Daten verschiedener Realtionen werden über den Dateninhalt selbst hergestellt. So könnte man etwa eine Adresse über die Kunden-Nummer aus einer zweiten Relation herausfinden:

Kunden-Nr.	PLZ	Ort	Strasse
6001	6000	Frankfurt	Mainzer Landstr. 120
6007	6100	Darmstadt	Heidelberger Landstr. 194

Die erste relationale Datenbank – zunächst für Systeme mit 8, später mit 16 und 32 Bit – war DBASE II. Diese Datenbank des Softwarehauses Ashton-Tate erlebte einen regelrechten Siegeszug durch die verschiedensten Anwenderkreise und wird heute noch genauso wie der Nachfolger DBASE III häufig benutzt. Auch für den ATARI ST wird eine Anpassung von DBASE II angeboten.

Für die Popularität relationaler Datenbanken sorgen zwei Eigenschaften, die sie von anderen Datenbankprogrammen unterscheiden.

- Einfachheit und Schnelligkeit, mit der Dateistrukturen definiert, benutzt und geändert werden können.



- Eine hohe Flexibilität im Datenzugriff ohne starre Zugriffspfade.

Adimens ST: Bedienungskomfort durch eigene Shell

Adimens ST ist eine solche relationale Datenbank. Das Programm wurde von dem renommierten deutschen Softwarehaus ADI entwickelt. Mit ihm kann man bis zu 16 unterschiedliche logische Dateien miteinander verknüpfen. Das erspart ein mehrfaches Erfassen und Ändern von Daten. Adimens verfügt über eine eigene „Shell", über die einzelne Programm-Module aufgerufen werden können. Das Programmpaket besteht aus folgenden Bestandteilen:

EXEC
Hauptprogramm plus Shell

INIT
Initialisierung der Datenbank

DRC
Erzeugung des Resource-Files für die Maske

EDITOR
Möglichkeit zum Einbinden eines beliebigen ASCII-Editors

REORG
Reorganisation der Datenbank

DOS-SHELL
Schnittstelle zum Kommandointerpreter

Erstellen einer Datenbank

Zuerst sollte man einen beliebigen Texteditor, z. B. 1st_Word.Prg, in Editor. Prg umbenenen und dann die benötigten Fiels auf die Sicherheitskopie der Adimens-Diskette kopieren. So sind alle Voraussetzungen für die Arbeit mit Adimens ST erfüllt.

Danach muß das Hauptprogramm EXEC aufgerufen werden, um von dort aus in alle notwendigen Programm-Module zu gelangen. Zuerst wählt man den Editor, um die Datenbankbeschreibung zu erstellen. Hier werden alle nötigen Vereinbarungen zur Erstelung der Datenbank (geplante Anzahl der Datensätze und Felder sowie Sortier- und Verbindungsschlüssel) getroffen. (siehe Tabelle 1)

Adimens ST stellt folgende Datentypen zur Verfügung:

c – alphanumerische Zeichen (max. 70)

i – numerische Ziffern Integer (max. 4)

l – numerische Ziffern Long-Integer (max. 12)

d – Dezimalzahlen (max. 8 incl. Punkt)

s – Betrag in kaufmännischer Notation (max. 14)

t – Datum (TT.MM.JJ)

```
.BASIS ADITEST = Adress, Telefon;

.DATEI Adress (1000);
       Name                              = c  20: Adress, Telefon;
       Vorname                           = c  20: Adress, Telefon;
       Rufname                           = c  10;
       Straße                            = c  30: Adress;
       PLZ                               = i   4: Adress;
       Ort                               = c  20: Adress;
       Geburtstag                        = t   8: Adress;
       Skip                              = c  12: Adress;Name;Vorname

.DATEI Telefon (1000);
       Name                              = c  20: Adress, Telefon;
       Vorname                           = c  20: Adress, Telefon;
       Rufname                           = c  15;
       1.Vorwahl                         = l  10: Telefon;
       1.Ortswahl                        = l  10: Adress;
       2.Name                            = c  20;
       2.Vorwahl                         = l  10: Telefon;
       2.Ortswahl                        = l  10: Adress;
       Notiz                             = c  70;
```

Tabelle 1

Wenn die Datenbankbeschreibung fertig ist, kehrt man ins EXEC-Programm zurück und ruft als nächstes das INIT-Modul auf, in dem man die eigentliche Maske generiert und die Dateien festlegt.

Dazu können die einzelnen Bestandteile der Maske beliebig auf dem Bildschirm – sogar über mehrere Bildschirmseiten (max. 64 Seiten pro Datenbasis!) – positioniert werden. Zudem ist es in diesem Programmteil möglich, eine bereits erstellte Datenbasis zu erweitern.

Allerdings sollte man darauf achten, bei der Positionierung nicht über bereits definierte Maskenteile zu fahren, da diese dann vom Bildschirm gelöscht werden. Die „überfahrenen" Maskenteile sind trotzdem im Speicher vorhanden und man kann Sie leicht wieder sichtbar machen. Lästig ist es dennoch.

Als nächstes Programm-Modul muß DRC aufgerufen werden. Es erzeugt aus den bisherigen Angaben ein Resource-File. Damit ist die Maskenerstellung abgeschlossen: Ab sofort kann mit der Datenbank gearbeitet werden.

Das Programm-Modul REORG kann bereits erstellte Dateien, deren Format zu klein geworden ist, nachträglich verändern. Dabei spielt es keine Rolle, ob es sich um zu klein geratene Felder oder um neue Felder handelt.

Als zusätzliches Modul wird der Kommando-Interpreter des ATARI-Entwicklungspakets mitgeliefert. Er wird einfach durch den Menüpunkt DOS-SHELL aufgerufen.

Das Arbeiten mit dem EXEC-Modul

Das EXEC-Modul ist, im Gegensatz zu den anderen Modulen, voll in GEM eingebunden. Mit ihm ist das Bearbeiten der Daten und die Verzweigung zu den anderen Modulen möglich.

Unter Bearbeiten versteht man das Erfassen, Ändern, Übertragen und Ausgeben von Datensätzen. Außerdem werden mit seiner Hilfe Datenbankinhalte nach bestimmten Gesichtspunkten sortiert und anschließend auf Diskette, Drucker oder Bildschirm ausgegeben. Drei Sortierroutinen stehen zur Verfügung: Man kann aufsteigend, absteigend und sequentiell sortieren. Das gewählte Sortierverfahren kann auf Diskette abgespeichert werden und wird dann automatisch beim Laden des EXEC-Moduls mit eingeladen.

Man kann nach den in der Datenbankbeschreibung definierten Schlüsselfeldern beliebig sortieren oder auch die Daten miteinander verknüpfen und diese Sortierdefinitionen sogar zum weiteren Gebrauch abspeichern. Wie Bild 1 zeigt, sind eine Vielzahl von Verknüpfungen möglich. Durch bloßes Anklicken der gewünschten Maskenteile und Verknüpfungssymbole werden die Wahlfelder generiert. Schön ist, daß man jegliche Arten von Beziehungen unter den Datenfeldern herstellen und sich regelrecht durch die Daten „durchklicken" kann. Das Relationale an dieser Datenbank ist wirklich optimal gelöst. Leider fehlt eine eigene Programmiersprache wie bei DBASE, doch die meisten Funktionen können durch entsprechende Definition der Schlüsselfelder ersetzt werden.

Jede Datei, die zu einer Datenbasis gehört, bekommt ein eigenes Karteikasten-

```
Desk  Daten  Wahl  Schalter  Optionen  Font  Programme
(Entwickler = 'adi?') und (Programmname =

Programmname: adimens st
  Kategorie1:
  Kategorie2:                                GEM-Appl. j/n: _
Kurzbeschreibung:

  Entwickler: adi?
 Distributor:

 OK  ABBRUCH  ZURÜCK   = # < > ≤ ≥ & |   ◇ SEITE  SEITE ◇

F1 EINGABE  F2 LÖSCHEN  F3 ÄNDERN  F4 SUCHEN  F5 ANZEIGEN  F6 DATEI  F7 SCHLÜSSL  F8 DATEIINF  F9 DB INFO  F10 ENDE
```

Bild 1: Die Verknüpfungsmöglichkeiten

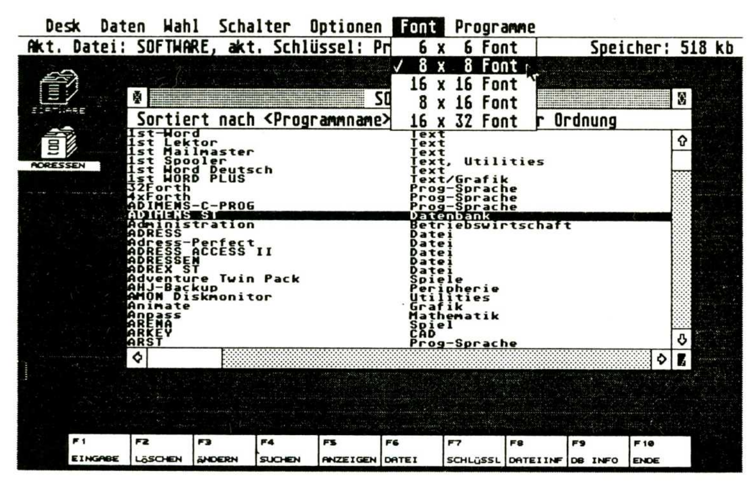


Bild 2: Ausgabe einer Liste

symbol auf dem Bildschirm. Durch Anklicken des Symbols wählt man zwischen den verschiedenen Dateien und läßt deren Inhalte anzeigen. Die Ausgabe der Daten erfolgt wahlweise in Listenform (siehe Bild 2) – wobei zwischen fünf verschiedenen Zeichensätzen gewählt werden kann – oder als Maske (siehe Bild 3). Wählt man die Listenform, erhält man durch Anklicken des betreffenden Listeneintrags sofort die dazugehörige Maske. Ein Nachteil daß jeder Dateizugriffspfad nur einmal angesprochen werden kann. Der Vergleich von verschiedenen Datensätzen einer Datei ist deshalb manchmal umständlich. Beim Eingeben der Daten in die Maske darf man nur das letzte Datenfeld der Maske mit Return abschließen.

Suchen mit „Wildcards"

Die Suche orientiert sich an beliebigen Schlüsselfeldern. Bei mehreren in Frage kommenden Datensätzen besteht die Möglichkeit, vor- oder zurückzublättern. Störend: Am Ende der Datensätze eines Suchkriteriums kann man nicht mehr zurückblättern, sondern muß von neuem beginnen.

Der Anwender kann auch Platzhalter, „Wildcards" genannt, einsetzen. So werden etwa bei einem Namen nur die ersten Buchstaben und danach ein „?" eingegeben, wobei Groß- oder Kleinschreibung egal ist. Adimens ST sucht nun die passenden Kombinationsmöglichkeiten und gibt sie als Liste oder Maske aus.

```
Desk Daten Wahl Schalter Optionen Font Programme
Akt. Datei: SOFTWARE, akt. Schlüssel: Programmname          Speicher: 518 kb

Programmname◇ ADIMENS ST__________________
  Kategorie1: Datenbank___________________
  Kategorie2: ____________________________         GEM-Appl. j/n: j

Kurzbeschreibung:
Echt relationale Datenbank in voller GEM-Oberfläche. Max. je 16 MB für
Daten und Schlüssel. Merkmalsarten: alphanum., (long) Integer, Dezim._
(Float. point oder Geldnotat.), europ. Datum. Wildcard-Suche_________
(?,!,$,#), Auswahl incl. UND, ODER, UNGLEICH. Sortierschlüssel________
innerhalb und Verbindungsschlüssel zu anderen Dateien (Verzweigung bei
Online-Recherchen oder Reports. Gestaltbare Direktausgabe auf Disk,___
Bildschirm, Drucker. Import, Export. Shell zu bel. Text (Serienbriefe)

  Entwickler◇ ADI_____________________
 Distributor◇ ATARI___________________

[VOR] [ABBRUCH] [ZURÜCK]                         [◇ SEITE] [SEITE ◇]
F1 EINGABE | F2 LÖSCHEN | F3 ÄNDERN | F4 SUCHEN | F5 ANZEIGEN | F6 DATEI | F7 SCHLÜSSL | F8 DATEIINF | F9 DB INFO | F10 ENDE
```

Bild 3: Beispiel einer Maske

Folgende „Wildcards" sind möglich:

„?"
vertritt einen String (mehrere Zeichen oder Wörter)

„!"
steht für ein beliebiges Zeichen

„#"
steht für eine Zahl

„$"
vertritt einen Buchstaben

Desktop beim EXEC

Das EXEC-Desktop ist ähnlich wie das Atari-Desktop aufgebaut: Ein zusätzlicher Pluspunkt für den ungeübten Atari-User. Die Funktionstasten sind in 1st_Word-Manier belegt.

Übrigens läuft Adimens ST in mittlerer oder in hoher Auflösung. Zwischen den beiden Möglichkeiten ist sogar vollständiger Datenaustausch gewährleistet: Ohne umständliche Anpassung der Maske kann weitergearbeitet werden.

Austauschen von Daten

Mit den Funktionen Import und Export lassen sich Daten mit anderen Systemen, etwa DB Master, austauschen. Meist sind jedoch kleine Datenkonvertierungsroutinen notwendig. Eine Routine zum Datenaustausch mit DB Master kann man sich beim Erwerb von Adimens beim Händler kopieren.

Ferner verfügt Adimens über eine Option namens „Mischen". Mit ihr kann man Daten zum Erstellen von Formularen, Serienbriefen und ähnlichen Texten abspeichern. Ein Beispiel dafür ist das Erstellen von Serienbriefen mit 1st_Word, wobei bei Datumsfeldern sogar eine automatische Anpassung mit Einfügen der Punkte zwischen Tag, Monat und Jahr geschieht.

Update-Version im Frühjahr

Für wen die Art der Maskengenerierung unter TOS zu umständlich ist, ist Trost in Aussicht: Zur CEBIT im Frühjahr wird ein Update vorgestellt werden, in dem das INIT-Modul in GEM eingebunden ist. Die Maskenerstellung geschieht dann ähnlich wie unter DB Master. Ferner werden in dieser Version Möglichkeiten zum Stapelbetrieb und zu Rechenoperationen implementiert sein. Die Rechenfelder sollen dann, ähnlich wie beim Menüpunkt „Wahl" definiert und abgespeichert werden kön-

können, so daß man immer darauf zurückgreifen kann. Die Update-Version wird gegen eine Gebühr im Austausch erhältlich sein, so daß man schon jetzt ohne Probleme mit Adimens arbeiten und seine „alten" Dateien später weiterbenutzen kann.

Ebenfalls für's Frühjahr ist eine Datenkonvertierung von DBASE- auf Adimens-Dateien geplant.

Passend zu Adimens ST ist schon jetzt eine C-Programmierschnittstelle zum Einbinden von Daten erhältlich. Die nötigen Lizenzen zur Nutzung in anderen eigenen Programmen können für DM 2000,– erworben werden.

Adimens ST wird für DM 499,– direkt von ATARI über die Fachhändler vertrieben.

ATARI Corp. (Deutschland) GmbH
Frankfurter STr. 89–91
6096 Raunheim

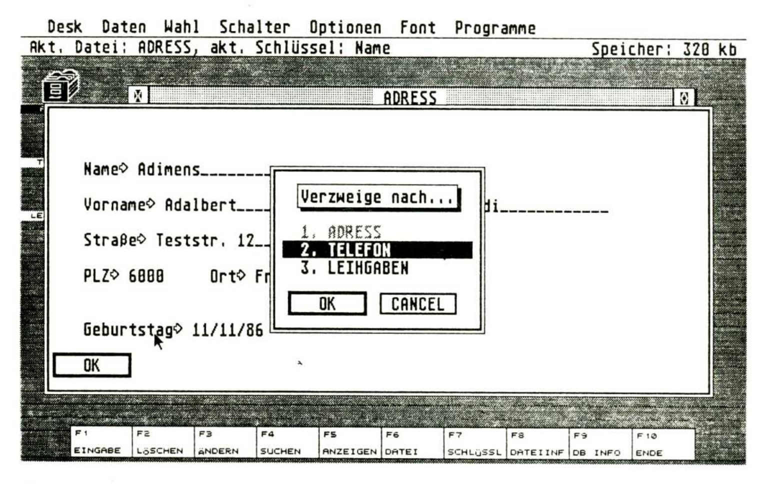


Bild 4: Verzweigen über Anklicken von Schlüsselfeldern

Fazit

Adimens ST hat uns sehr gut gefallen und ist allen ernsthaften Datenbankanwendern zu empfehlen. Allerdings sollte man bei intensiver Nutzung die anschaffung einer Harddisk erwägen.

+ gute, einfache Bedienung
+ frei definierbare Schlüsselfelder
+ vielfältige Verknüpfungsmöglichkeiten
+ hohe und mittlere Auflösung
+ Datenaustausch mit anderen Programmen
+ mehrere Datenbasen gleichzeitig bearbeitbar
+ Shell-geteuerte Bedienung der Module
+ schneller Datezugriff
+ GEM bei EXEC-Modul
+ Anschlußmöglichkeit einer C-Schnittstelle
+ „Wildcards" beim Suchen
+ Reorganisation der Datenbank möglich
+ gutes Preis-Leistungsverhältnis
+ deutsche Dokumentation
– Maskenerstellung mühsam
– Kein Zurückblättern nach Erreichen des letzten Datensatzes
– Zugriffspfad nur einmal anzusprechen
– Noch keine Rechenoperationen möglich

Maximale Kapazitäten von Adimens ST

16 (logische Dateien pro Datenbasis
64 Bildschirmmaskenseiten pro Datenbasis
32000 Datensätze in einer (logischen) Datei
4096 Zeichen pro Datensatz
150 gleichzeitig verknüpfbare Merkmale
512 Merkmale in einer Ausgabeliste
16 MBytes Umfang einer physikalischen Datei
16 MBytes Umfang einer physikalischen Schlüsseldatei

Klaus Heuer / HE

BIETE SOFTWARE

GRAFIKTERMINAL (Tektronix 4010, VT100-Komp.) durch Emulationsprogramm. Mit Assembler-Quellcode, vielfach bewährt. Info: Bossler, Germanenstr. 9, 7 Stuttgart 80

★ ★ Verkaufe DEGAS 105,- DM orig. mit Handbuch. 0851/51905

GfA-Basic u. Hamlet-Schach zusam. 180 DM. Tel. 069/391636

Public-Domain-Service für ST. Liste anfordern (3,–). A. Stein, Schillstr. 24, 4830 Gütersloh 1

Mindshadow VB 70,-. 07392/6178

?-BASIC in GFA-BASIC!
Erzeugt aus jedem **(auch C-64)** BASIC Ein-Befehls-Zeilen, erhält Sprungadressen, setzt ENDIF ein! Vorkasse 30,– DM. M. Prall, Isestr. 57, 2 Hamburg 13

PROFORTRAN 05723/3881

Für Lehrer, Prüfer usw.: **Testauswertung u. Namensdatei** Bewertung aller **Aufgaben** und **Schüler** mit Ausdruck. Diskette gegen 40 DM bei Helmut Geiger, Wiesenstr. 13, 7250 Leonberg 7

Megamax-C Lattice-C ST-Pascal+ alle mit Handbuch. 05641/8754

ATARI ST-Public Domain **jetzt direkt in Berlin.** 40 Disks erh. je DM 8,– frei Info Tel. 8254163

★ Images ★ Pictures ★ Bilder ★ Digi-Pics, 16 Grst S/W.D00/.Pi3 Gratisliste bei: T. Wägli ★ ★ ★ Winkelackerstr. 4 CH-4573 Lohn

Verkaufe **GST-C-Compiler** Orig. DM 200,– Tel. 09253/8213

Verkaufe folgende Originale: ST-Heimf., GfA-Basic Vers. 2, ST-Textd. Textomat, Datamat, BHILON Fast-Basic Compiler, Leader Board Tournament. 50 % unter EK. Tel.: 08031/5262

XXXXXXXXXXXXXXXXXX
X Public-Domain Software für X
X Atarie ST. Kostenlose Liste X
X bei PD-Club, Hasenwinkel 13 X
X 5778 Meschede (es lohnt sich) X
XXXXXXXXXXXXXXXXXX

Achtung Lehrer! Profi-Notenverwaltungsprogramm - GEM gesteuert für DM 40,– bei Seidl, Aschaffenbg.St. 12, 8520 Erlangen

PD-Software für den ST? Wir sind die Spezialisten mit dem größten Programmangebot in der BR Deutschland!!! Gratiskatalog bei R. Köhler, Mühlgasse 6, 6991 Igersheim. Es lohnt sich!

Lotto-Programm
20-seitige Dokumentation für 2,40 in Briefmarken, F. Klein, Dessauer Str. 5a, 89 Augsburg 21

★ ★ ★ CHART ★ ★ ★
Komfortables Wertpapierprogr. Viels. Chartdarstellung + Depotv. incl. GfA-RT-Interpr. Info frei Disk 99,– bei C. Weisenböhler Waldstr. 2, 7175 Vellberg

Lohnsteuerjahresausgleich / Einkommensteuer 86/GfA/ 20 DM incl. Disk, GfA-Interpr., Porto Vorkasse/J.Höfer/Grunewald 2a, 5272 Wipperfürth, Tel. 021923368

Software für Erwachsene! Info, schnell und diskret bei Soft, Pf. 1619, 4830 Gütersloh

Lohnsteuerberechnung für 1986? Auf Atari-ST kein Problem, DM 99,– Scheck an: A. Gödde, 5940 Lennestadt 11, H. Rumpstr. 15

Spiele, Spannung, Action und vieles mehr biete ich auf über 40 PD-Disks. Gratisinfo bei Ralf Markert, Balbachtalstr. 71, 6970 Lauda-Oberbalbach

ST-AKTIEN Prg., Depotverwalt. + Charterstellung, DM 90,– incl. Vorkasse als Scheck o. Schein, Info gratis, J. Schneider, Kaiserstr. 65, 6900 Heidelberg

★ Aktuellste PD-Software ★
Incl. Porto, Verp. und Disk, 1S ab 45 DM, 2S ab 5 DM. Info geg. frank. Rückumschlag. A. Teubert, Gleiwitzerstr. 27, 48 Bielefeld 1

C-Bibliothek: Quick-Sort, W.-tag zum Datum, etc.; 29 DM, Info 2,- H. Michels, 34 Gö, Föhrenweg 11

GFA-Super-Alert-box.
Schluß mit 4 Zeilen á 30 Zeichen. Maximal 19 Zeilen á 72 Zeichen. Aufruf wie Alert. 30,– V.-Scheck, B. Bartels, Pestalozzis. 16, 1 Bln. 12

Wärmebedarfs./KZahlberechnung
DIN4701/83 GEM Bedienerführung Dipl.-Ing. V. Koch, Am Mehnacker 11 3563 Dautphetal 3, Tel. 064687652

Wollten Sie nicht schon immer mal einen **Bundesligaverein** führen? Mit **FUSSBALL-MANAGER** spielen Sie um die **deutsche Meisterschaft** und den **Pokalsieg.** Kaufen und verkaufen Sie Spieler! Verbessern Sie die ärztliche Betreuung Ihrer Spieler! Fördern Sie den Nachwuchs und vieles mehr... Fragen Sie Ihren Händler nach diesem interessanten Strategiespiel von **Poffel Products,** Zollhausstr. 39, 6520 Worms 26. nur s/w für nur 49,– DM

● Public Domain Software ●
Riesiges Angebot, kopiere auch **einzelne** Programme, ein- und doppelseitig, uvm. **Nur 4,50 DM je 360 kB!** Info: Eckart Stinshoff / Hökholz / 2336 Waabs

KALENDER herstellen auf dem ATARI ST + Drucker. Das individuelle Geschenk mit Pfiff!!! Das Prg. bietet phantastische, äußerst vielfältige Bilder und was sonst so zu einem Kalender gehört. Info o. Best. (39,-- incl. Disk, Porto) b. S. Henrichsmeier, Oberdorf 58, 5305 Alfter 4

Public Domain Software
Hochwertige Software zu minimalen Preisen. Jede randvolle Disk nur 11,90 einseitig – oder 17,90 doppelseitig. Fordern Sie unseren Gratiskatalog und vergleichen Sie unser Angebot. Ing.-Büro Manfred Ohlms, Postfach 6312, 4400 Münster

PUBLIC DOMAIN PROGR.
auf Markendisk (10) nur 79,- DM zuzügl. Porto
Inge Dander, Rockefellerstr. 66 8 München 45 – Tel. 089-3111782

Megamax C, Mark Williams C, ST-PASCAL/Text/Platine, VIP, Zeitschr. / Bücher, Service Man. Tiefpreis Tel: 06131-475662

PD-SOFTWARE FÜR DEN ST!
Wir sind die Spezialisten mit dem größten Programmangebot in der BR Deutschland! Außerdem div. Soft + Hardware zu absoluten Tiefstpreisen! Gratisinfo bei Reinhold Köhler, Mühlgasse 6, 6991 Igersheim, postwendend!!!

Vokabel-Prg. für Atari ST zum Eing., Editieren, Suchen u. Lernen von Vok.; 59 DM, Info 2 DM H. Michels, 34 Gö, Föhrenweg 11

An alle ST-Nutzer! Warum teure Software kaufen wenn es doch für beinahe jede Anwendung auch ein PD-Programm fast zum Nulltarif gibt! Über 80 Disks, randvoll und nichts doppelt!!! Gratiskatalog anfordern bei Reinhold Köhler, Mühlgasse 6, 6991 Igersheim.

★ ★ ★ **Public-Domain** ★ ★ ★
```
PPPP  DDD
P   P D    D
P   P D     D -Software für
PPPP  D     D den Atari-ST
P     D     D
P     D    D
P     DDD
```
Wir haben **Super-PD-Software** zu einem **revolutionären Preis** (ab **2,50 DM** Kopierkosten/Disk). **Gratis**katalog anfordern: H. Blankenstein, Ettenhofen 31, 8031 Weßling

★ 50 Disk Public Domain Prg. ★ á 9,– DM NN zzgl. Porto/NN bei J. Dander · Rockefellerstr. 66 8000 München 45 · 089/3111782

★ ★ ★ Public-Domain ★ ★ ★
Superbillig! Paket: 5 doppelseitige Disk. **nur DM 40,–** 10 einseitige **DM 68,–.** Liste bei: Blankenstein, Ettenhofen 31, 8031 Weßling, Tel. 08153/1623 nach 14 Uhr

PD-Software für Atari ST in Unmengen vorhanden. Mehr als 80 randvolle Disketten, spottbillig. Hunderte von Programmen stehen zur Auswahl. Da ist garantiert für jeden etwas dabei. Gratiskatalog anfordern bei Reinhold Köhler, Mühlgasse 6, 6991 Igersheim!!!

Beste PD Software für den ST. Nur **4 DM pro Disk!** Info gegen Rückumschlag bei T. Horak, Baumeisterweg 12, 7 Stuttgart-1, T. 0711853921

Wärmebedarfs./KZahlberechnung
DIN4701/83 GEM Bedienerführung Dipl.-Ing. V. Koch, Am Mehnacker 11 3563 Dautphetal 3, Tel. 064687652

PD-Software für den ST? 1000 Programme haben wir, und es werden täglich mehr!!! Gratiskatalog bei Reinhold Köhler, Mühlgasse 6, 6991 Igersheim!

PUBLIC DOMAIN AUS KALIFORNIEN!
Natürlich von IDL
(siehe Anzeige)

BIETE HARDWARE

20 MB-Festplatte DM 1700,– SC 1224 (ATARI-Color-Monitor) für DM 950,– Tel. 069/473547

Typenraddrucker Quen-Data dwp 1120 für Atari ST mit Traktor NP 1800 DM für 1200,– DM zu verkaufen VB Telefon ab 20 Uhr 0571/24703

Floppy SF 314 m. Zubehör neuw. VB 530,– 02473/8407 17–20 Uhr

SF354 DM 200. Tel. 0212/819412

TV/AV-Modulatoren 150 DM
1 Megabyte steckbar 225 DM mit Marken-Ram's, gesockelt Info bei: Stachowiak-Computer 4300 Essen · Burggrafenstraße 88

SF314 neu VB 650 DM. SF 354 VB 250,-. Tel. 0531/353628 Dirk

SF354 neu VB 250,- 02331/72081

Atari ST520+, SM124, SF314, Software Mo–Do ab 19⁰⁰ 07259/1685

Verkaufe **SF 314 / 519,– DM !** neuwertig (1/2 Jahr) gebr.! **Doppellaufw.** je 1 Meg, 1A Qual. (Eigenbau) anschlußfertig **870,–** Seitz Ch., Untere Hagenstr. 16, 8562 Hersbruck / **Zugreifen!** /

■ SF314 500,–, SF354 300,– ■
■ M. de Wijs, T. 004161872728 ■

SF354 DM 260 Tel. 06743/1631

Verkaufe SF 354 für 200,– DM!!!
★★★ 06898 37979 ★★★

Drucker-/Scartkabel 4 m lang; SF314: 500,-; **Fremdfloppy** 300,-; Orig.-Software; Tel: 0201-466588

SF 314, neuw., 600 DM 089-8571915

Atari ST 520+ incl. 1 Floppy SF354 + SF314 mit Bildschirm SM124 mit Star NL10 Drucker + versch. Software zu verkaufen. Neupreis ca. DM 5600,- f. 4500,- VHB abzugeben. T. 06133/58679

S/W-Monitor SM124, GfA-Basic, Datamat, 1ST-Word: 02943/7242

Cumana Doppel-Floppy 2x720 kB 2 Mon. alt, 900 DM.
Tel. 0251/617159 (Gottschling)

Verkaufe SF 354 mit Netzteil + Kabel 364,– DM. 089/94903

Atari ST Eprommer m. Tex 139,- Uhr+Datum Accu gepuffert 35,- Fa. Bause, 2 Hmb 55, T. 040862771

SF354 neu orig. verp. DM 350,– Frense, Buchfinkweg 10, 6534 Stromberg, Tel. 06724/1304

★**Floppystecker** DM 19,95★
★**Floppykabel**, rund, 1 m★
★ ATARI-offenes Ende 28,– ★
★ **Anschlußfertige Kabel** ★
★ATARI-1X3,5" DM 39,90★
★ATARI-1X5,25" DM 39,90★
★ATARI-2X3,5" DM 59,90★
★**ATARI-2X5,25" DM 59,90**★
★**NEU:** FL.-KABEL **mit inte-**★
★**griertem Treiber** DM 69,– ★
★dto. für 2 Laufw. DM 89,– ★
★ Wir fertigen auch Kabel ★
★ nach Ihren Wünschen ★
★**Netzteil 5V/12V** DM 39,90★
★1MB **Laufwerk** NEC 1036A★
★ DM 299,– ★
★dto. **ATARI modifiziert** ★
★ DM 319,– ★
★ **Floppystationen,** integr. ★
★ Netzteil, anschlußfertig ★
★ ATARI-grau ★
★ Einzelstation DM 499,– ★
★ Doppelstation DM 869,– ★
★ **PAL-Interface** f. Farb TV ★
★DM 299,–/**ST Oszillograf**★
★mit **Sound Sampler** 439,– ★
★**1 MB Speichererw.** 219,– ★
★dto. **ohne Löten** 269,– ★
★**Epromkarte** 128 KB 89,– ★
★ **Monitorstecker** DM 11,90 ★
★**Monitorkabel** f. Monochr. ★
★ Fremdmonitor DM 39,90 ★
★ **Scartkabel** 2m DM 49,90 ★
★**Druckerkabel** 2m DM 34,90★
★ **Public D. Disk's** können ★
★ Sie sich bei uns selbst ★
★ zusammenstellen / Liste ★
★**gegen DM 1,90 Rückporto**★
★**MIWIKO Computertechnik**★
Mesteroth 9 · 4250 BOTTROP 2
TELEFON: 0 20 45 / 8 16 38
oder ab 17⁰⁰ Uhr 0 20 43 / 7 13 74

SF354 neuw. VB 289,- 06029/8509

VIDEO DIGITIZER ST PRO neuw. DM 698,– 06121/86703

SF314 480,– 0 61 81 / 7 19 20

SM124, 400 DM, Tel. 089/661769

10 3,5" Disketten f. 50,- inkl. Nk (Vorausk.) Fa. Bandt, Altenbergring7, 6991 Igersheim, 07931/42117

PLATINE ST orig. VHB 300,– (neu 698,– DM) 05461/2557

■ DISKETTEN ■
■ 5 1/4", 48 tpi, **DM 0,99**, 2D ■
■ 3 1/2", 135 tpi, **DM 3,19**, 1DD ■
■ 3" Schneider **DM 5,85** ■
■ auch andere, bes. Garantie ■
■ Allg. Austro-AG, Ringstr. 10 ■
■ D-8057 Eching, Tel: 08133/6116 ■

OKI ML 192+1ST-Word (ST-Anpass.) + 10 Prg. / 1048 DM / 07236-6516

Wehrpflichtiger verkauft: **Atari 260: 500 DM**, SF354: 200 DM, **Disk+Programm** (PD, Forth, Pascal, Spiele...): **4,20 DM** je Disk. M. Horak, Baumeisterweg 12 7000 Stuttgart 1
T. (Samstags ab 8h): 0711/853921

ST-13 D. Fl.: 699 DM
ST-16 D. Floppy: 3.5 + 5.25 Zoll: 898 DM; auch andere Modelle; kpl. anschlußfertig!
FD 1036 A nur 240,– NT!!!
★★★★★★★★★★★★★★
ST-17: 2x3 1/2" + 1x5 1/4"
6★85 TR = 2.61 MB Formt.!
in Gehäuse + Rinkern–Netzt.
Betr.bereit: 1.255,00 DM
Tel: 0 61 51 / 5 13 95

Sprachausgabe auf dem ST! Info anfordern von: Schlegel, Schwarzachstr. 3 7940 Riedlingen, 07371/2317

SUCHE SOFTWARE

Suche Fortran-Comp., CAD/Zeichen Prog., Spiele, etc. 02238/52795

Taxiabrechnungsprogramm für Atari ST gesucht, oder Hinweis woher so etwas bezogen werden kann. Chiffre 0187H

Suche für ST+ Programme Fachrichtung Mabau, T. 02330/10623

Suche orig. Software, Midisoftw., Bücher. Preul, Steilshooperstr. 183, 2000 Hamburg 60, 040/6919454

★★★★ SCHWEIZ ★★★★
Suche **Megamax C-Compiler** mit **deutschem Handbuch.** L. Weber Hirscheng. 35, 5416 Kirchdorf, CH / Tel. 056/825844

Suche Anleitung f. Graphic Artist + Easy-Draw gegen Finderlohn. Graf, Laubenweg 26, 4300 Essen 1

SUCHE HARDWARE

Kaufe SF 354 max. DM 150,– Kippenberger, Mühlenstr. 6, 44 MS

Suche Cobol-Compiler. C.Grandke Hofmannstr. 89, 6103 Griesheim

TAUSCH

Zuverlässiger ST-Partner sucht Prg. aller Art, nur Originale im Tausch, oder ST-CLUB n. Köln Uwe Petersen, Urnenstr. 26 B 5000 Köln 80 (Dellbrück)

KONTAKTE

ST Kontakte gesucht 05171/3523

■**Suche ST-Anwender aus CH**■
Progr.- & Erfahrungsaustausch.
Tel: 042/36 74 27 (ZG, 14 Jahre)

VERSCHIEDENES

ATARI wählen per Computer!!! Interf.+Softw. 49,-. T. 0911/575036

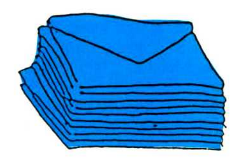
Atari ST Public Domain Service Für nur 17 DM die neuesten PD Progr. Info gegen Rückumschlag Rolf Rieckmann, Hauptstr. 33 3101 Langlingen

ST-Computer-Zeitschr. Jahrg. '86 komplett 45,– DM (05345) 771

Feb. '87 — Anzeigenschluß am 6. Januar 1987

März '87 — Anzeigenschluß am 3. Februar 1987

Verspätete Einsendungen kommen in die darauffolgende Ausgabe!



Herzlichen Glückwunsch!
Die ST-Redaktion kann bald Ihren ersten „Geburtstag" feiern. Aus diesem Anlaß würde ich es begrüßen, wenn man in der Januar-Ausgabe eine Leserbefragung zum Inhalt und der Gestaltung der Zeitschrift machen würde.
Vor etwa einem Monat habe ich die Aufmachung Ihres Magazins stark kritisiert, damals mit der Oktober-Ausgabe vor Augen. Jetzt liegt mir die November-Ausgabe vor, die ich für sehr gelungen halte.

Die ST-Computer hat sich inzwischen bei allen ST-Anwendern und Freaks als Fachblatt etabliert. In der Hoffnung, daß die Berichterstattung noch **aktueller** wird, läßt sich der Zeitschrift eine rosige Zukunft vorhersagen.

Ulf Blanke, 3127 Brome-Altendorf

Antwort: Vielen Dank für Ihren Glückwunsch, es freut uns immer wenn sich zwischen den vielen fachlichen Anfragen auch einmal ein persönliches Wort verbirgt.

Sehr geehrte Redaktion!
Für die künftige Themenauswahl möchte ich Ihnen eine kleine Anregung geben.
Seit der Erstausgabe nehmen Berichte und Folgen zum Betriebssystem und zu Programmiersprachen einen weiten Raum ein. Vorhandene Literatur dürfte dadurch anschaulich ergänzt werden und Programmierern zusätzliche Anregungen liefern.
Anders sieht es jedoch für den Nuranwender aus. Zu der in letzter Zeit erschienenen professionell nutzbaren Basissoftware (Textverarbeitung und Datenbankprogramm) fehlt es an begleitender Literatur und vergleichenden Gegenüberstellungen in Computerzeitschriften.
Hier sollte dem interessierten Anwender eine aktuelle und ausführliche Berichterstattung (z. B. über Adimens, Megabase, DB-Expert) angeboten werden.
Der hohe Bedarf an dieser ST angemessenen Software und ihre zu erwartende weite Verbreitung lassen sicher auch eine Rubrik mit speziellen, von den Experten der Redaktion eingebrachten, Tips und Tricks beim Leser und geschäftlichen ST-Anwender der Anklang finden.
Zum Schluß möchte ich Sie bitten, Ihren Einfluß bei Software-Häusern geltend zu machen, damit deren Programme die Grafikfähigkeiten der 24-Nadeldrucker auch nutzen. Ein Drucker wie der NEC P6 mit seiner Auflösung von 360V ★ 360H Pkte./Zoll sollte nicht nur für Textverarbeitung bestimmt sein. Wettbewerbsvorteile liegen doch hier klar auf der Hand.

Klaus Braun, 4390 Gladbeck

Antwort: Die Themenwahl einer Zeitschrift ist immer kompliziert, da viele Interessengruppen abgedeckt werden müssen. Ihren Vorschlag einer Tips & Tricks-Ecke liegt uns auch sehr am Herzen. Deshalb wollen wir alle Leser auffordern, uns ihre Erfahrungen und Erkenntnisse mitzuteilen, da nur so ein breites Spektrum abgedeckt werden kann.

Ich wohne in der österreichischen Hauptstadt der Steiermark, Graz. Können Sie mir in dieser Gegend einen Computerclub nennen, der sich mit dem ST befaßt?

Oliver Nußbaum, Graz

Antwort: Soeben flattert uns ein Brief aus Graz, auch bekannt als Hauptstadt der Steiermark, auf den Tisch. Eine ST-Gesprächsrunde trifft sich jeden ersten Mittwoch im Monat bei einer gemütlichen Bier- oder Apfelsaftrunde beim Sternwirt in der Münzgrabenstraße. Kontaktadresse: Günter Jernej, Rosenhaingasse 7, 8010 Graz

Leser antworten Lesern

In dieser Rubrik können Sie, liebe Leser, unsere Zeitschrift mitgestalten. Jeder, der Lust und Kenntnis hat, kann seine Frage oder Antwort an uns schicken. Wir werden sie dann veröffentlichen.

Anschluß von Fremdfloppies am ST
Nachdem ich mir im Mai 86 einen Atari 260 ST mit einem Laufwerk SF 354 gekauft habe, faßte ich nun den Entschluß ein doppelseitiges Laufwerk zusätzlich anzuschließen. In der Zwischenzeit hatte ich aufmerksam Ihre Artikel zum Anschluß anderer Floppies an den ST gelsen. Da ich noch ein 2★80 Spur 3 1/2 Zoll Laufwerk von meinem Sinclair-Spectrum besaß, faßte ich den Entschluß dieses zusätzlich anzuschließen. Ich nahm die SF 354 auseinander und verlängerte den Shugart-Bus. Auf der Platine, wo sich die runde Eingangsbuchse befindet, wurde eine Brücke von Pin 6 nach Pin 12 Shugart-Buchse eingelötet, da sonst das Select-Signal für das 2. Laufwerk fehlt. Außerdem entfernte ich die Widerstandsmatrix aus dem zusätzlichen Laufwerk. Bisher konnte ich noch keine fehlerhaften Funktionen an dem zusätzlichen Laufwerk feststellen. Bei dem Laufwerk handelt es sich um ein 3 1/2 Zoll Laufwerk von Misubishi Typ MF 353-12 M.

Herbert Schubert, 4370 Marl

Plotter am Atari ST
Wer hat bereits Erfahrungen mit einem Plotter (DIN A3 oder größer) gemacht. Welche Probleme gibt es bei dem Betrieb? Schnittstellen, Software, Service, u.a.? Ich besitze einen Atari 260 ST+.
Rainer Schröer, 4600 Dortmund 50

Vorschau Februar '87

Das können Sie voraussichtlich in der nächsten Ausgabe lesen:

- **Der Macintosh-Emulator**
  Was leistet er wirklich? Was wird zusätzlich benötigt? Ein Vergleichstest mit dem Original Apple Macintosh.
- **Art & Film Director**
  Mit diesen beiden Programmen ist Grafikanimation in Vollendung möglich. Nach langem Warten sind sie endlich auch in Deutschland erhältlich.
- **Profimat ST**
  Mit dem Profimat ST ist nun der zweite GEM-eingebundene Assembler auf den Markt gekommen. Wir testen ihn für Sie.
- **Funktionseingabe in GFA-BASIC**
  Auf einfache Art und Weise wird die Eingabe von Funktionen in Dialogform gelöst. Wie, daß erfahren Sie in der nächsten Ausgabe.

Die Februar-Ausgabe erscheint am 30.01.87

Inserentenverzeichnis

Unser Preisausschreiben aus ST-Computer 10/86!

Die richtige Antwort lautete: „mit dem Finger im Einkaufsführer"

Die Gewinner sind:

1. Preis: Andreas Schmidt, Pforzheim
Ein Hamlet Schach

2.–6. Preis: Markus Posch, Salzburg/Österreich
Mathias Fröhlich, Sigdorf
Hans-W. Kothe, Marburg
Peter Mrosek, Marl-Sinsen
Erik Rieger, Bruckberg
Je ein GfA-Basic Buch aus dem Heim-Verlag incl. Programmdiskette

7.–9. Preis: Dr. K. Riedel, Steinfurt
Mike Hirsch, Altlußheim
Thorsten Bomberg, Borken
Je ein Buch aus dem Heim-Verlag inclusive Programmdiskette

10.–12. Preis: Alexander M. Arensberg, Schmallenberg
Joachim Moritz, Bamberg
M. Heinemann, Kiel
je eine Low-Cost Uhr als Bausatz

13.–30. Preis: Karl-Heinz Michiels, Schwalmtal
Meyer, Barnstorf
Sara Cuylen, Krefeld
Daniel Pfiffner, Baar/Schweiz
Heinz Grothenn, Delmenhorst
Gerhard J. Morgenbeßer, Wilhelmshaven
Isolde Weiss, Karlsruhe-Durlach
Ralf Höhne, Hamburg
Rainer Prengel, Celle
Oliver Grimm, Flensburg
C.-J. Kammler, Wülfrath
Klaus Ohlenmacher, Idstein
Hagen Noll, Frankfurt
Dirk Hmaann, Idar-Oberstein
Thomas Langen, Berlin
Hubert Mundinger, Leonberg
Achim Borgs, Duisburg
Romain Poulles, Vichten/Luxembourg
Je eine Programmdiskette ST-Computer Nov./Dez. 1986

Impressum

ST-Computer

Herausgeber: Heim Fachverlag, Heidelberger Landstraße 194, 6100 Darmstadt 13, Telefon (0 61 51) 5 60 57

Verlagsleitung: Hans-Jörg Heim

Redaktion: Uwe Bärtels (UB-Chefredakteur). Harald Egel (HE) Marcelo Merino (MM), Harald Schneider (HS) Schwalbacher Str. 64, 6236 Eschborn, Tel. 0 61 96/48 21 58,

ST-Computer
Redaktion, Postfach 59 69, 6236 Eschborn

Redaktionelle Mitarbeiter: Markus Nerding (MN), Oliver Joppich (OJ), Uli Eickmann (UE) Jürgen Leonhard (JL)

Autoren dieser Ausgabe: K. Heuer, S. Schuler, T. Weinstein, Dr. K. Sarnow, B. Biewer, K. Tödter, P. Gebhart

Titelseite: M. FABIAN

Produktion: Klaus Schultheis (Ltg.), Bernd Failer, Susanne Failer, Bela Kumar, Carola Schwarze, Martina Simacek

Anzeigenverkaufsleitung: Uwe Heim

Anzeigenpreise: nach Preisliste Nr. 2 gültig ab 1.7.86

Vertrieb: Hans-Jörg Heim, Uwe Heim, Heide Schultheis

Erscheinungsweise: 11 x jährlich

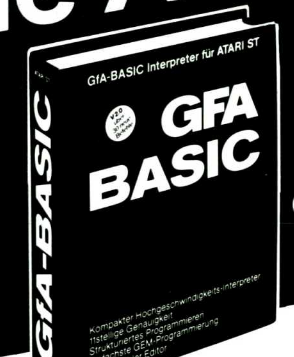
Bezugspreis: Einzelheft DM 6,–. Jahresabonnement DM 60,– inklusive der gesetzlichen Mehrwertsteuer und den Zustellgebühren für 11 Ausgaben. Ausland: 80,– DM inkl. Versand

Bezugsmöglichkeiten: ATARI-Fachhändler, Zeitschriftenhandel, Kauf- und Warenhäuser oder direkt beim Verlag unter obiger Adresse.

Druck: Ferling Druck Darmstadt

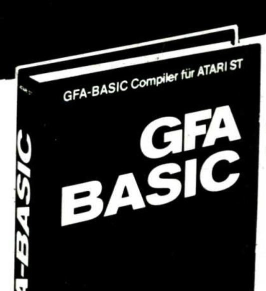
Manuskripteinsendungen: Programmlistings, Bauanleitungen und Manuskripte werden von der Redaktion gerne angenommen. Sie müssen frei von Rechten Dritter sein. Mit ihrer Einsendung gibt der Verfasser die Zustimmung zum Abdruck und der Vervielfältigung auf Datenträgern im Heim Verlag. Honorare nach Vereinbarung. Für unverlangt eingesandte Manuskripte wird keine Haftung übernommen.

Für alle ATARI ST



Kompakter Hochgeschwindigkeits - Interpreter,
11stellige Genauigkeit,
strukturiertes Programmieren,
einfachste GEM-Programmierung,
komfortabler Editor.

GFA-BASIC Interpreter V 2.0
DM 169,–



GFA-BASIC Compiler
DM 169,–

Schneller 2-Pass-Compiler, benötigt keinen Linker,
erzeugt sehr schnelle, kurze Programme ohne Runtime-Modul,
voll kompatibel zum Interpreter, einfachste Bedienung.

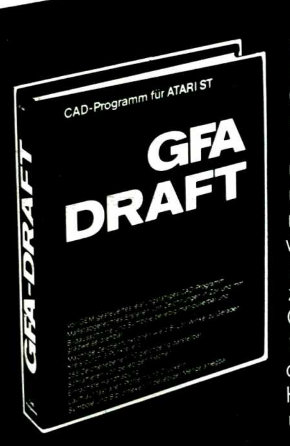
Alle bisherigen Preise und konditionen verlieren ihre Gültigkeit

3D-Animationsprogramm für ATARI ST
GFA VEKTOR
GFA-VEKTOR

Schnelles, interaktives 3D-Grafik-Programm zum Generieren von 2D- und 3D-Objekten.

Aus GFA-BASIC heraus können problemlos 2D- oder 3D-Spiele, Animationen oder bewegte Simulationen erzeugt werden.

GFA-VEKTOR DM 149,–



Voll GEM-gesteuertes, leistungsfähiges CAD-Programm, maßstabsgerechtes Erstellen von Zeichnungen in Zoll und mm, Bildausschnitte und Symbole beliebig manipulierbar und gradweise drehbar, mächtige Zeichenfunktionen wie z. B. Lot, Winkel zu Geraden, Kreis durch 3 Punkte, etc. 255 Zeichenebenen (10 gleichzeitig darstellbar). Schraffieren von beliebigen Flächen. Einfachste Handhabung. Läuft auf fast allen Plottern und Druckern. Symbole und Bibliotheken in beliebiger Menge anlegbar.

GFA-DRAFT DM 298,–

...Anruf genügt: 02 11-58 80 11

GFA Systemtechnik GmbH

Heerdter Sandberg 30
D-4000 Düsseldorf 11
Telefon 02 11/58 80 11

GFA SYSTEMTECHNIK